Multi-Button IR Remote Control using the MC68HC908LT8

Designer Reference Manual

M68HC08
Microcontrollers

DRM082
Rev. 0
09/2006

freescale.com

freescale™
semiconductor

Multi-Button IR Remote Control using the MC68HC908LT8

Designer Reference Manual

by: T.C. Lun
Freescale Semiconductor, Inc.
Hong Kong

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify that you have the latest information available, refer to http://www.freescale.com

The following revision history table summarizes changes contained in this document. For your convenience, the page number designators have been linked to the appropriate location.

Revision History

Date	Revision Level	Description	Page Number(s)
September, 2006	0	Initial release	N/A

Table of Contents

Appendix A
Schematic

Appendix B
Program Listing

Chapter 1
Introduction

1.1 Introduction

This document describes a reference design for an infrared (IR) remote control (RC) solution using the MC68HC908LT8 microcontroller.

For many air conditioner and small home appliance applications, there is a need for a wireless user interface such as a remote control unit to send data from a transmitter to a receiver using infrared communication. The basic requirements of an IR remote control unit are: lower power consumption in standby mode; low operating voltage; low system cost; and easy code modification for customizing to different models.

This reference design includes both the transmitter and the receiver unit. A feature of this reference design is a 16-pin MON08 programming interface header for in-circuit Flash programming and debugging in the remote control transmitter and receiver units.

1.2 Freescale's MC68HC908LT8 Low-Cost MCU

The MC68HC08LT8 is a member of the low-cost, high-performance M68HC08 family of 8-bit microcontroller units (MCUs). All MCUs in the family use the enhanced M68HC08 central processor unit (CPU08) and are available with a variety of modules, memory sizes and types, and package types.

Features include:

- High-performance M68HC08 architecture
- Fully upward-compatible object code with M6805, M146805, and M68HC05 families
- Low-power design; fully static with stop and wait modes
- Maximum internal bus frequency:
  - 4-MHz at 5-V operating voltage
  - 2-MHz at 3-V operating voltage
- Dual oscillator module
  - 32.768-kHz crystal oscillator
  - 1- to 16-MHz crystal oscillator
- 8,192 bytes user Flash memory
- 128 bytes of on-chip random-access memory (RAM)
- Two 16-bit, 2-channel timer interface modules (TIM1 and TIM2) with selectable input capture, output compare, pulse-width modulation (PWM) capability on each channel
- Programmable periodic interrupt (PPI)
- 4/3 backplanes and static with maximum 24/25 frontplanes liquid crystal display (LCD) driver

- Up to 38 general-purpose input/output (I/O) ports:
  - 4 keyboard interrupts with internal pull up
  - 2 × 15 mA high-current sink pins
- System protection features:
  - Optional computer operating properly (COP) reset
  - Optional low-voltage detection with reset and selectable trip points for 3-V and 5-V operation
  - Illegal opcode detection with reset
  - Illegal address detection with reset
- Master reset pin with internal pull-up and power-on reset
- $\overline{\text{IRQ}}$ with schmitt-trigger input and programmable pull up

1.3 Reference Demo Board

The remote control reference design has the following features:

- Transmitter unit: MC68HC908LT8 controlled 9-button with LCD
- Receiver unit: MC68HC908LT8 controlled 2-button with LCD and LED indicators
- 38-kHz carrier frequency generated by software delay
- Easy re-programming and debugging by 16-pin MON08 interface
- Low operating voltage down to 1.8 V
- Low power consumption in standby mode, typically 20 μA[1]

Figure 1-1 shows the transmitter and receiver unit of the IR remote control reference design.

1. The power consumption is dependant on application and system requirements. The 20 μA assumes that all modules are turned off except PPI, LED, KBI modules, and subsystem clock (32.768 kHz crystal).

(a) MC68HC908LT8 IR RC Receiver

(b) MC68HC908LT8 IR RC Transmitter

Figure 1-1. Infrared Remote Control Reference Design

Chapter 2
Fundamentals of IR Remote Control Communication

2.1 Configuration of the IR Remote Control Unit

An IR remote control transmitter generates infrared rays to a receiver by way of a digital control frame pattern. The infrared transmitting diode and the infrared receiving module are important components for an efficient IR transmission through air. The carrier frequency for home appliance applications is typically around 38 kHz.

A typical configuration of IR remote control is shown in Figure 2-1.

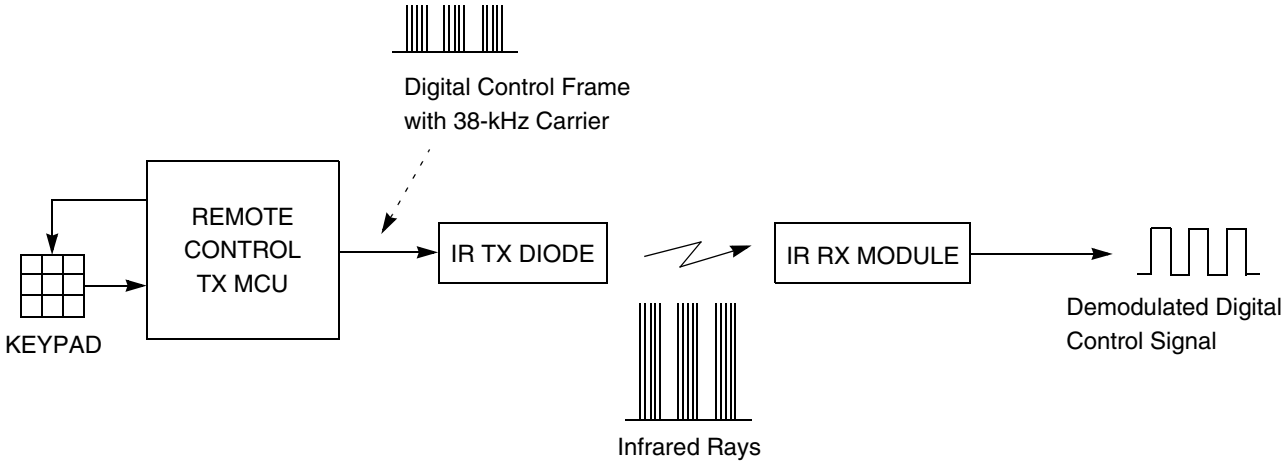


Figure 2-1. Configuration of IR Remote Control Unit

2.2 Control Frame Format

The IR control frame pattern is specific for different transmitter-receiver designs. It depends on application requirements such as controller purpose and features. Figure 2-2 shows the typical example of the control frame waveform that is used in this IR remote control reference design.

In Figure 2-2, the carrier is the 38 kHz with a 1/3 duty cycle. Having IR transmitting diode using 38-kHz carrier and 1/3 duty cycle allows a low power design for the IR transmission. If the carrier was 1/2 duty, the transmitting diode will be on for 13 μs and off for 13 μs. But for 1/3 duty, the diode is on for 8 μs and off for 18 μs. A reduction in turn-on time means a reduction in power consumption.

The data bit for 0 or 1 is based on the duration of the carrier on/off. For data 0, both carrier on and off times are 0.5 ms. For data 1, the carrier on time is 0.5 ms and the carrier off time is 1.5 ms.

Typically, the data frame consists of the header code, several bytes of data code, one byte of customer code, and one stop bit. The header code is used to indicate to the IR receiver that following transmissions will be the data code and customer code. The data code is used for control purposes, such as on/off, increase/decrease, modes, etc. The customer code is used for identifying different customers. And, the stop bit is to indicate it is the last bit of the current transmission.

In this reference design, the above frame format is used for an air conditioner remote control unit.

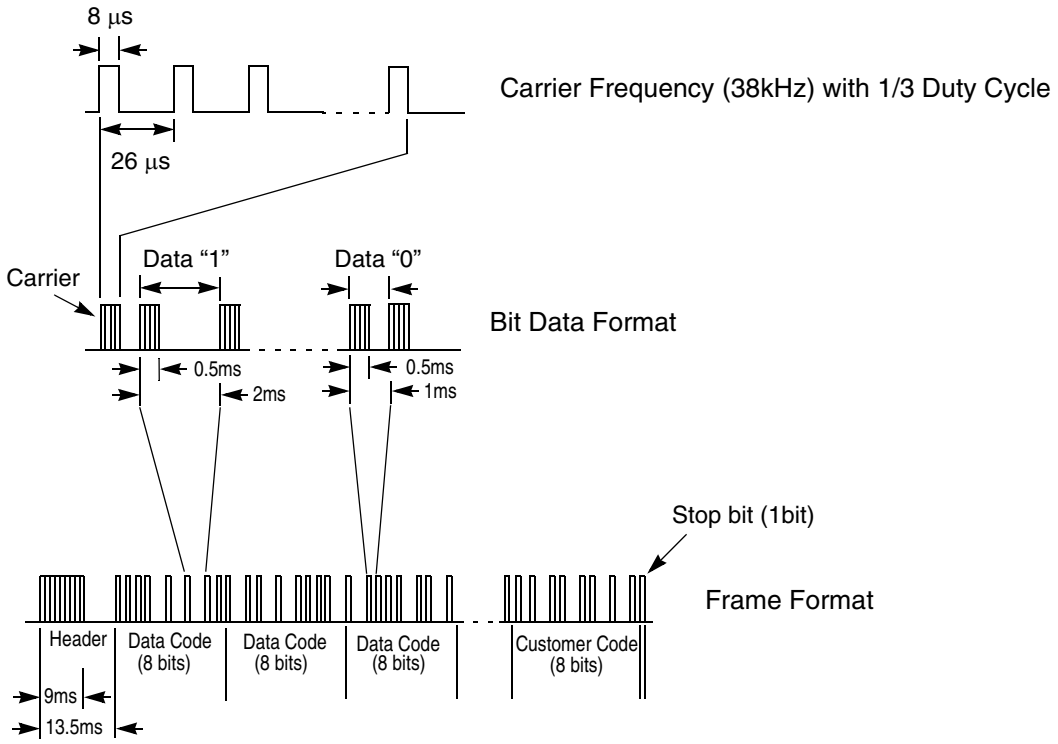


Figure 2-2. Control Frame Waveform

Chapter 3
System Concept

3.1 System Specification

This reference design demonstrates a remote controller for air conditioner/small appliance applications with re-programming and debugging features. The design meets the following performance specifications:

- Low power consumption in standby mode (transmitter unit)
- Low operating voltage (transmitter unit)
- 16-pin MON08 interface for software development (transmitter and receiver units)
- MC68HC908LT8 controlled transmitter and receiver for system evaluation in real time
- Transmitter and receiver uses standard type AAA batteries as power source

Figure 3-1(a) shows the front of the transmitter unit with the 9-key keypad. Figure 3-1(b) shows the back of the transmitter unit with the BDM interface header and battery cover.

Figure 3-2(a) shows the front of the receiver unit with the key switch, LCD and LED display, and the IR receiver module. Figure 3-2(b) shows the back of the receiver unit with the MON08 interface header, battery holder, and ON/OFF switch.

3.2 Application Description

The design uses a MC68HC908LT8 in both the transmitter and the receiver unit.

In the transmitter unit, the MC68HC908LT8 performs the following tasks:

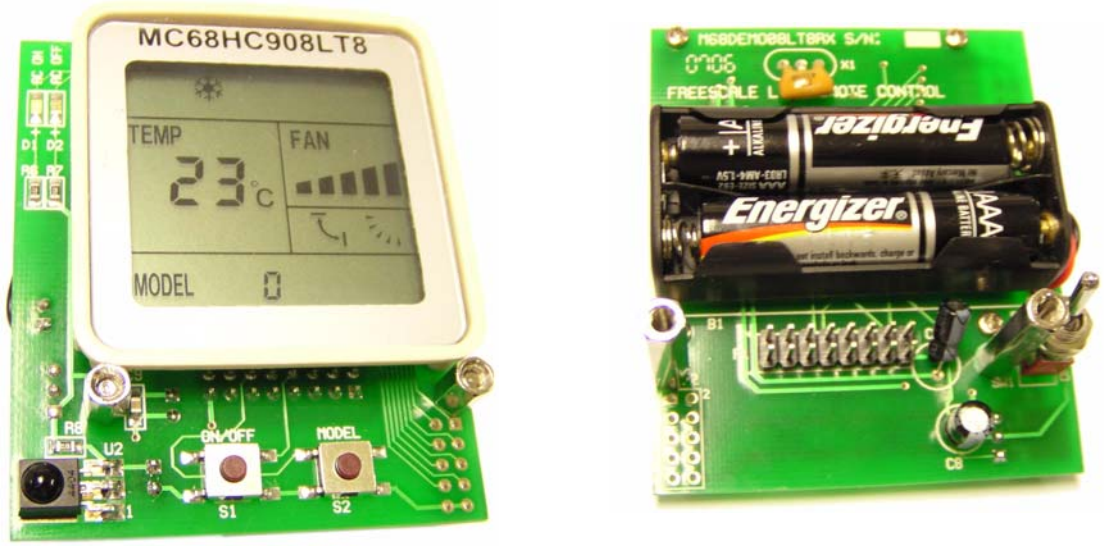
- Keyboard scanning
- Frame encoding
- Carrier generating
- Transmitting the encoded frame to IR with carrier
- LCD driving

In the receiver unit, the MC68HC908LT8 performs the following tasks:

- Keyboard scanning
- Frame decoding
- LCD and LED displaying

(a) Front of IR RC Transmitter (b) Back of IR RC Transmitter

Figure 3-1. MC9RS08KA2 IR Remote Control Transmitter Unit



(a) Front of IR RC Receiver (b) Back of IR RC Receiver

Figure 3-2. MC68HC908LT8 IR Remote Control Receiver Unit

3.3 Control Process

Since the design is targeted for an air conditioner remote controller application, some general control parameters must be included. For example, power ON/OFF, temperature data, and mode selection.

Table 3-1 summarizes the control data frame definition for this reference design.

NOTE

Since each customer has their own requirements and definitions, Table 3-1 only includes the general and common control parameters. Additional parameters can be added, thus increasing the frame length by the additional control bytes.

Table 3-1. Remote Control Frame Definition

Data Code Name	Bit Definition								Function	Remarks		
	7	6	5	4	3	2	1	0				
C1	0								A/C OFF			
	1								A/C ON			
		0	0	0					AUTO mode(1)	no temp.		no sleep
		0	0	1					COOL mode(2)	custom temp.	custom wind	
		0	1	0					HUMIDITY mode(2)	custom temp.	custom wind	
		0	1	1					WIND mode(2)	custom temp.	custom wind	no sleep
		1	0	0					HEAT mode(3)	custom temp.	custom wind	
					0	0			°C			
					0	1			Reserved			
					1	0			°F (Lower range)			
					1	1			°F (Higher range)			
							0		Light ON			
							1		Light OFF			
								X	Reserved			
C2					0				Sleep OFF			
					1				Sleep ON			
						0			Swing OFF			
						1			Swing ON			
							0	0	AUTO Wind Speed			
							0	1	LOW Wind Speed			
							1	0	MIDDLE Wind Speed			
							1	1	HIGH Wind Speed			

Continued on next page

Table 3-1. Remote Control Frame Definition (Continued)

Data Code Name	Bit Definition								Function	Remarks		
	7	6	5	4	3	2	1	0				
									Temperature	°C	°F	°F
										C1[3:2] = 0:0	C1[3:2] = 1:0	C1[3:2] = 1:1
	0	0	0	0						15°C	59°F	75°F
	0	0	0	1						16°C	60°F	76°F
	0	0	1	0						17°C	61°F	77°F
	0	0	1	1						18°C	62°F	78°F
	0	1	0	0						19°C	63°F	79°F
	0	1	0	1						20°C	64°F	80°F
	0	1	1	0						21°C	65°F	81°F
	0	1	1	1						22°C	66°F	82°F
	1	0	0	0						23°C	67°F	83°F
	1	0	0	1						24°C	68°F	84°F
	1	0	1	0						25°C	69°F	85°F
	1	0	1	1						26°C	70°F	86°F
	1	1	0	0						27°C	71°F	
	1	1	0	1						28°C	72°F	
	1	1	1	0						29°C	73°F	
	1	1	1	1						30°C	74°F	
Model[4]												
C3	0	0	0	0					**0**			
	0	0	0	1					**1**			
	0	0	1	0					**2**			
	0	0	1	1					**3**			
	0	1	0	0					**4**			
	0	1	0	1					**5**			
	0	1	1	0					**6**			
	0	1	1	1					**7**			
	1	0	0	0					**8**			
	1	0	0	1					**9**			
					0				**Model Set ON**			
					1				**Model Set OFF**			
						x	x	x	**Reserve**			
C4	1	0	1	0	1	0	0	1	**Customer Code[4]**	Same model number between transmitter and receiver		

NOTES:
1. Default mode for the reference design after a power-on-reset.
2. Default value of temperature is 25°C and needs to store temperature and wind speed individually.
3. Default value of temperature is 28°C and needs to store its temperature and wind speed individually.
4. Same model and customer code for transmitter and receiver.

Chapter 4
Hardware

4.1 Hardware Implementation

This chapter will focus on the hardware implementation of transmitter and receiver units.

The IR remote control transmitter and receiver units can be divided into the following parts:

- Oscillator circuit
- Keypad scan
- IR transmitter diode drive (TX only)
- IR receiver (RX only)
- LCD and LED display
- MON08 interface

4.2 MC68HC908LT8 IR Remote Control Transmitter

The MC68HC908LT8 IR remote control transmitter unit is mounted on an optimized PCB and fits in an actual remote controller casing, with keypad, LCD, battery holder, and a MON08 interface header for firmware development and system evaluation.

This reference design uses the 52-pin packaged MC68HC908LT8 to implement the basic functions of the IR remote control transmitter unit.

4.2.1 Oscillator Circuit

Since the MC68HC(9)08LT8 MCU is designed for remote control applications, it has two independent clock drives. Both transmitter and receiver units have the same oscillator circuit, as shown in Figure 4-1. The reason for the two clock sources is due to the low power requirements in standby mode of remote control applications. For a remote control transmitter unit, the unit is in standby mode for the majority of the time. The unit wakes up only when it detects a key press. Therefore, in this standby mode, the main bus clock in the MC68HC(9)08LT8 can be stopped and the slower subsystem clock is used to drive the LCD display and the programmable periodic interrupt (PPI) module.

Figure 4-1 shows the common crystal oscillator circuit for HC08 family MCUs. The component values shown are optimized for the MC68HC(9)08LT8, which is the same for both the transmitter and receiver units. The 4-MHz clock is divided by four for a MCU bus of 1-MHz. A slower bus speed will further lower MCU power consumption in run mode.

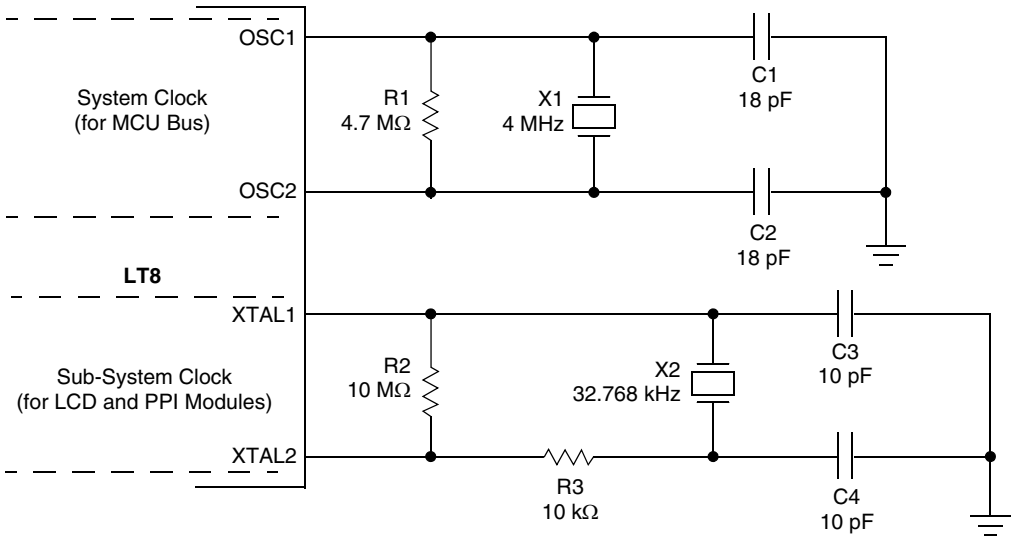


Figure 4-1. Oscillator Circuit for MC68HC908LT8

4.2.2 Keypad Scanning

The transmitter unit has a 9-button keypad, while the receiver unit has only two.

4.2.2.1 Transmitter Unit

The MC68HC(9)08LT8 has four I/O pins with keyboard interrupt (KBI) capability. Having these KBI pins means the MC68HC(9)08LT8 can be put into stop mode for power saving when no buttons are being pressed.

The nine buttons on the transmitter unit are configured in a 3-by-3 matrix for key scanning. Shown in Figure 4-2, pins KBI0–KBI2 and PTA4–PTA6 are used to form a key matrix. Before entering standby mode, PTA4–PTA6 are set to output low and KBI0–KBI2 enabled for keyboard interrupts. Pressing any button from SW1–SW9 will wakeup the MC68HC(9)08LT8 from standby mode. Once out of standby mode, the button is debounced and decoded.

A detailed description of keyboard scanning is discussed in the 5.3 Transmitter Software Implementation.

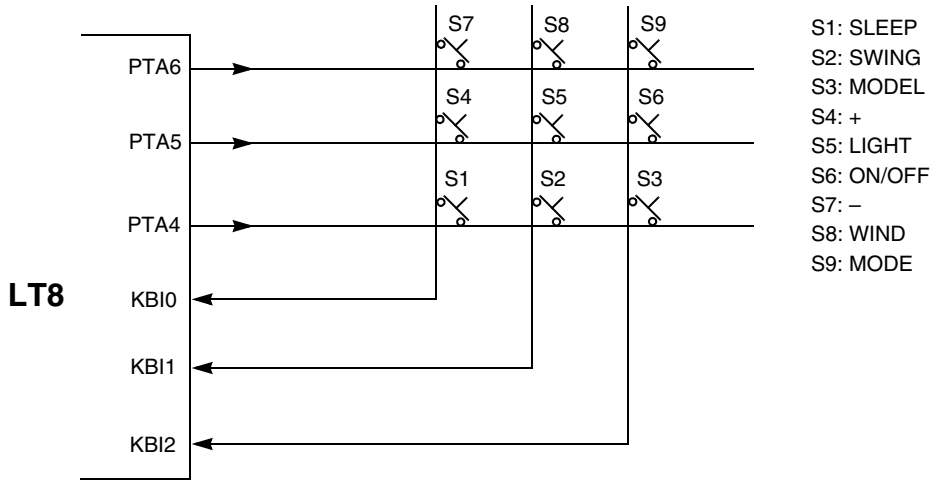


Figure 4-2. Transmitter 3-by-3 Keyboard Matrix

Table 4-1 summaries the function or meaning of each button on the transmitter unit:

Table 4-1. Buttons on the IR Remote Control Transmitter Unit

Button	Function
S1	This is the sleep mode button. Pressing S1 activates the sleep timer and turn off the receiver LCD. The air conditioner switches off when the sleep timer expires. The actual sleep timer is not implemented on this reference design.
S2	This is the air swing selection button. Pressing S2 toggles the air conditioner louver air swing on and off. The corresponding icon on the transmitter/receiver LCD is activated accordingly (see 4.3 Transmitter LCD and LED Display).
S3	This is the model selection button. With the first power on the transmitter, the model iron will flash to let the user select the model number. If the user wants to change the model number, he/she can press S4 “+” or S7 “–” key to increase or decrease the model number while the model iron is flashing. Or, the user can press the S3 “model” key to choose the desired model number that is shown on the LCD display. If the transmitter is powered on a second or later time, the model number can only be changed if the user presses the model key. Then, the model iron will flash again to let the customer change the model number by pressing the “+” or “–” key.
S4	Button S4: This is the “+” or increase button for temperature or model number depending on the condition. • If the model key is not pressed (i.e., model icon is not flashing), pressing the “+” key will increase the temperature by 1. Or, pressing the “+” key for more than 5 seconds will increase the temperature faster and continuously until the “+” key is released. Additionally, during the temperature change, the control data frame will transmit out after the “+” key is released. • If the model key is pressed (i.e., model icon is flashing), pressing the “+” key will increase the model number by 1. Or, pressing the “+” key more than 5 seconds will increase the model number faster and continuously until the “+” key is released. In this case, no control data frames will be transmitted after the “+” key is released.
S5	This is the LCD backlight ON/OFF button. Pressing S5 toggles the backlight on the transmitter/receiver LCD on and off. In this reference design, this button actually toggles one of the receiver LCD icons on and off.
S6	This is the ON/OFF button. Pressing S1 toggles the air conditioner power on and off. When the receiver is in the OFF mode, the OFF LED will be on (ON LED is off) and the LCD will be off. When the receiver is in the ON mode (ON LED is on) OFF LED is off and the LCD will be on.
S7	Button S7: This is the “–” or decrease button for temperature or model number depending on the condition. • If the model key is not pressed (i.e., model icon is not flashing), pressing the “–” key will decrease the temperature by 1. Or, pressing the “–” key for more than 5 seconds will decrease the temperature faster and continuously until the “–” key is released. Additionally, during the temperature change, the control data frame will transmit out after the “–” key is released. • If the model key is pressed (i.e., model icon is flashing), pressing the “–” key will decrease the model number by 1. Or, pressing the “–” key more than 5 seconds will decrease the model number faster and continuously until the “–” key is released. In this case, no control data frames will be transmitted after the “–” key is released.
S8	This is the WIND speed selection button. Pressing S3 toggles through the wind speeds of the air conditioner: AUTO → LOW → MIDDLE → HIGH and back again to AUTO (see Table 3-1. Remote Control Frame Definition). The default setting is AUTO when the air conditioner is switched from off to on. The corresponding icon on the receiver LCD is activated accordingly (see 4.3 Transmitter LCD and LED Display).
S9	This is the mode selection button. Pressing S9 toggles through the operating modes of the air conditioner: AUTO → COOL → HUMIDITY → WIND → HEAT and back again to AUTO (see Table 3-1. Remote Control Frame Definition). The corresponding icon on the receiver LCD is activated accordingly (see 4.3 Transmitter LCD and LED Display).

4.2.2.2 Receiver Unit

There is no need for power saving in the receiver, because the power in the air conditioner is coming from the AC main. Therefore, there is also no need to force the system (receiver) into standby mode. For this reference design, we used key polling instead of the keyboard interrupt. Figure 4-3 shows the connection of two keys. The keys are connected to PTA0, PTA1, and ground and use the polling technique to check which key has been pressed. In addition, both PTA0 and PTA1 are needed to enable the internal pullup to maintain the logic high when no key has been pressed. If any key is pressed, PTA0 or PTA1 will sense the logic low. There are different methods found between the transmitter and receiver due to their system requirements.

There are two keys in the receiver:

- S1 is the air conditioner ON/OFF control button. The receiver will turn ON or OFF depending on the ON/OFF control bit coming from the transmitter, or if S1 is in the receiver itself.
- S2 is the MODEL selection in the receiver. The model number will be increased by one for each S2 pressed. And, it will be changed from 0 to 9 and then changed back to model 0 again. The model number between transmitter and receiver must be the same for valid communication. If there are different model numbers between them, the receiver will neglect the command from the transmitter.

NOTE

After powering on the receiver, the model number in the receiver should be 0 by default. And, the transmitter must be set as model 0 for valid communication.

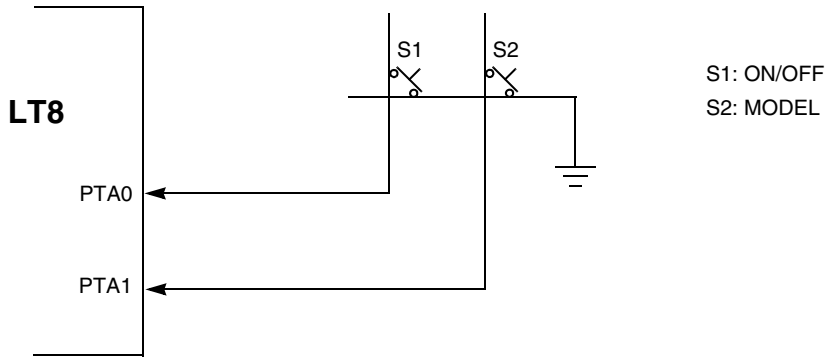


Figure 4-3. Receiver Key (Polling)

4.3 Transmitter LCD and LED Displays

In the transmitter, the LCD display only shows the air conditioner status after the last control command from the transmitter. In the receiver, the LCD display shows either the command from the transmitter control data frame or the key command from the receiver itself. So, two additional LEDs are needed to show the status of the receiver (air conditioner), such as power on and power off.

Figure 4-4 shows the LED and LCD display on the MC68HC908LT8 IR remote control receiver unit.

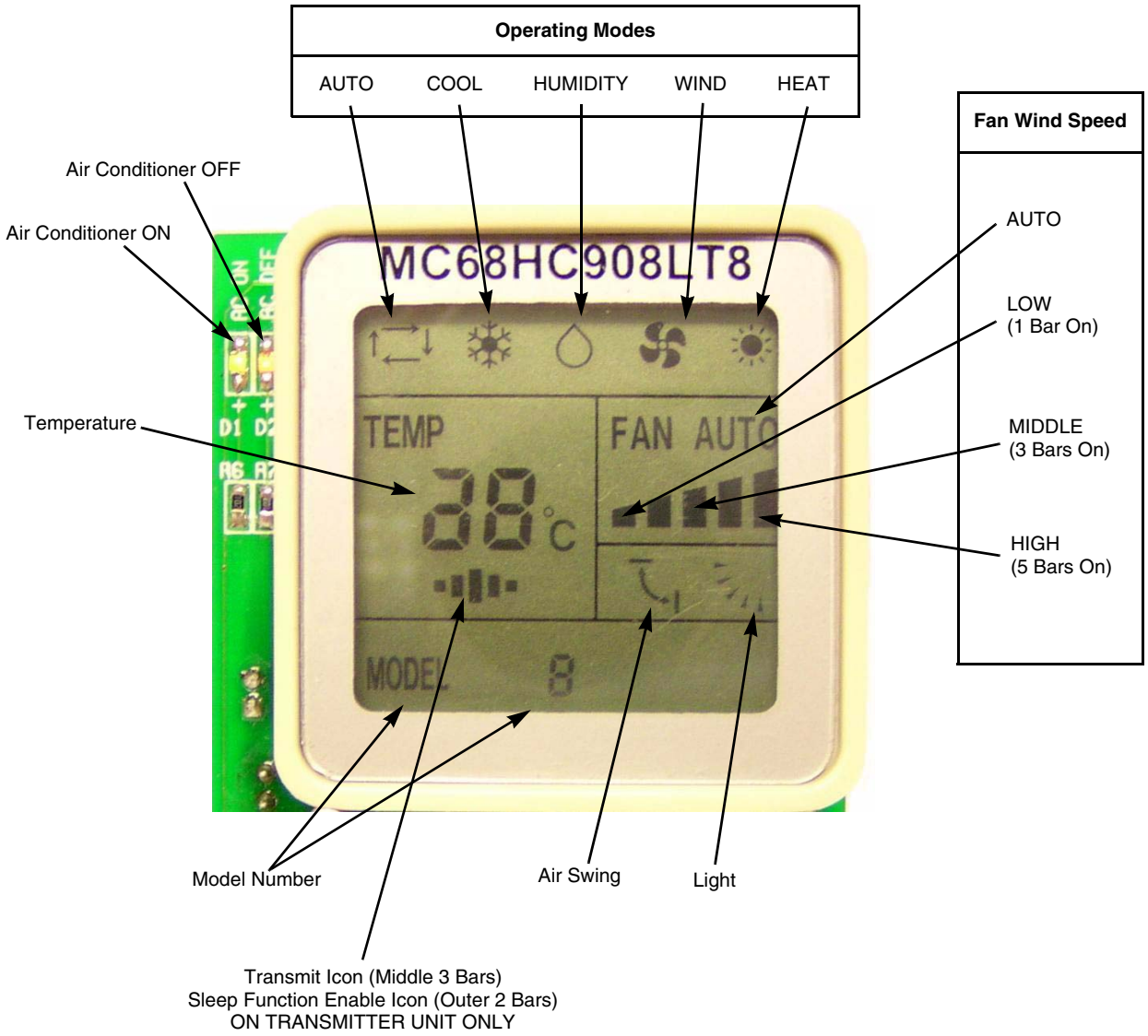


Figure 4-4. Typical Air Conditioner Receiver LCD Display

4.4 IR Transmitter Diode Drive

Taking system cost into consideration, the MC68HC(9)08LT8 MCU drives the IR transmitting diode directly. Figure 4-5 shows a typical drive circuit for an IR transmitting diode.

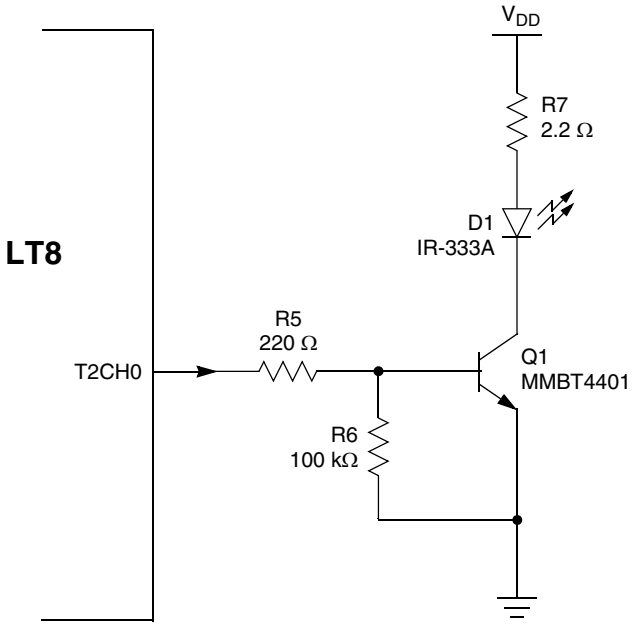


Figure 4-5. Circuit for IR Transmitting Diode

In the above circuit, the timer channel T2CH0 is used to drive the IR transmitting diode. The timer channel will generate the 1/3 duty cycle 38-kHz waveform. Using the timer to generate the PWM waveform has several advantages such as:

- Simply control flow
- Small code size
- Easier re-configuration of duty cycle and frequency of the carrier

The resistor R6 is used to force the transistor Q1 to an OFF state during the system power up stage. And, the resistor R7 is used to limit the current of the IR transmitting diode. The value of R7 is dependent upon the requirement of output power in the IR transmitting diode. Lower value on R7 will increase the output power of the IR transmitting diode. On the other hand, the duty cycle of PWM can also change the output power of the IR transmitting diode.

4.5 Infrared Receiving Module

Since the IR receiver board is used only to demonstrate the ability of the MC68HC(9)08LT8 IR remote control transmitter, the operating voltage of the receiver board is 3 V. The common operating voltage of a receiving board in an air conditioner is 5 V. So, we selected a IR receiving module that will also work with an operating voltage of 3 V. Therefore. the sensitivity of the receiving module may be somewhat different if used in a 5-V IR receiving module.

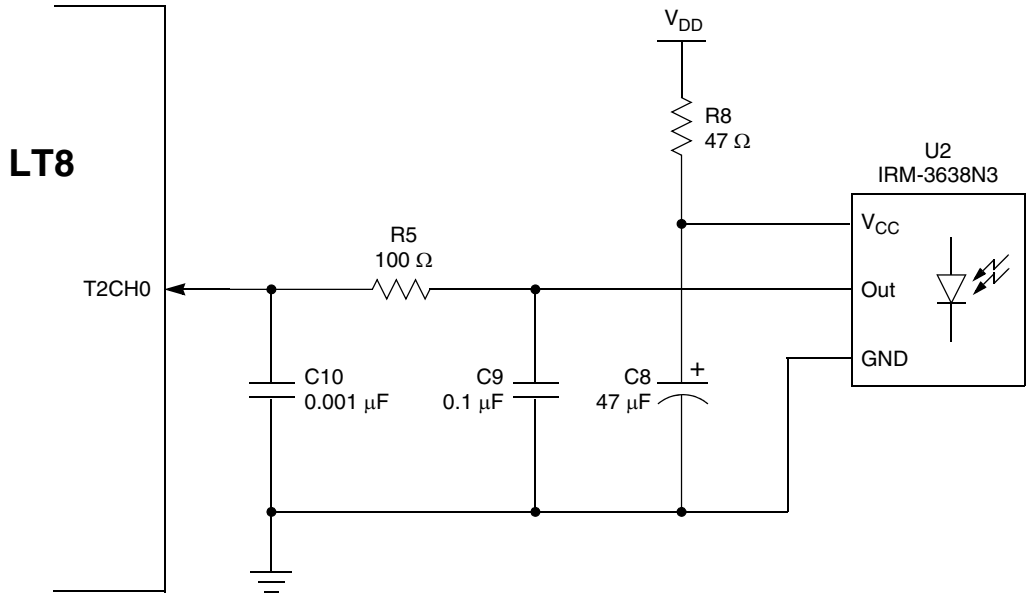


Figure 4-6. Circuit for IR Receiving Module

Figure 4-6 shows a typical IR receiving module circuit with MCU. In the above circuit, both the MCU and the IR receiving module work at 3-V V_{DD}. When U2 senses a 38-kHz waveform, it will demodulate it back to the base-band signal and drive it to the output pin “Out”. C8 and C9 are the bypass and decoupling capacitors for the U2 used to filter out any noise that coupled from the power supply source. R5 and C10 form the low-pass filter that filters out any unwanted high-frequency noise from U2. We used one of the MC68HC(9)08LT8 MCU timer channels, it configures as an input capture to decode the incoming base-band signal from the IR receiving module. Using timer channel T2CH0’s input capture can simplify the code, making it easier to decode the incoming signal.

In addition, there are some hints to help choose the IR transmitting diode and IR receiving module, such as the same center freakiness and same peak wavelength. The half angle is depended on the requirements of the transmitter and receiver. In normal cases, we may choose the same half angle in both the IR transmitting diode and the IR receiving module. On the other hand, the reception distance also needs to be considered.

4.6 MON08 Interface Header

For easier reprogramming of Flash and evaluating purposes, a 16-pin MON08 header is included in this reference design. The MON08 interface provides in-circuit programming and debugging features.

To help the user more easily re-program and evaluate the pair of demo(s) (IR remote control transmitter and IR remote control receiver), both of them have added the 16-pin standard MON08 interface. This interface provides the in-circuit programming feature for the user. The user can re-program the code into both the transmitter and the receiver to test their implementation. In addition, some hardware interface board is needed to connect between the PC and the MON08 header, such as P&E CyclonePro or HC08 low-cost tool (M68UICS08).

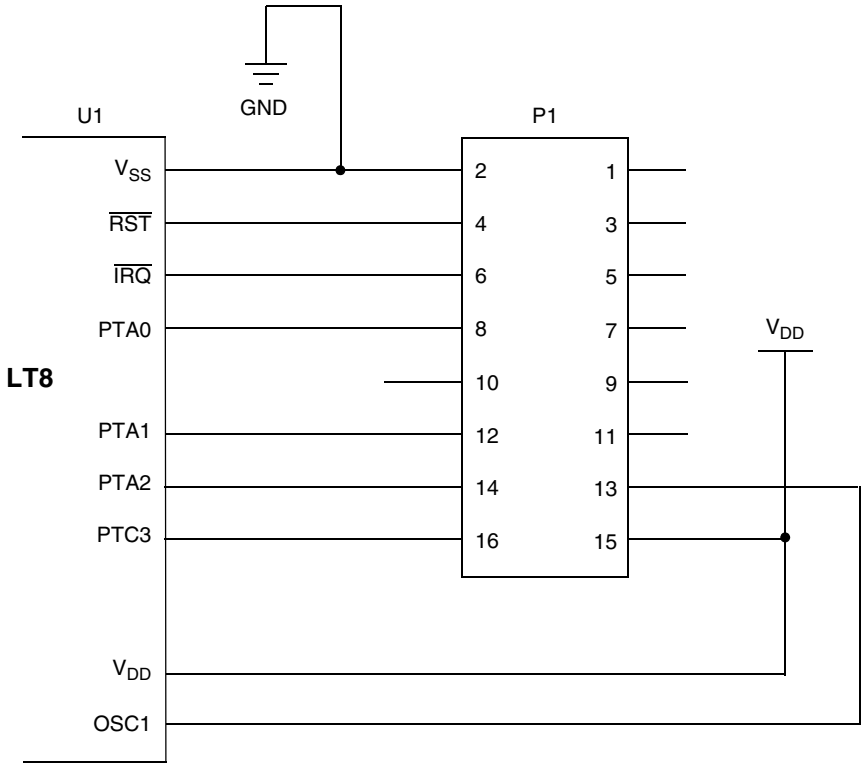


Figure 4-7. MON08 Interface Circuit

Figure 4-7 shows the connection between MON08 header with MC68HC(9)08LT8 MCU in both transmitter and receiver boards. When the user wants to re-program their code to the demo, they will need to connect the hardware interface board between the PC and the MON08 header. Then, program their code to the demo in order to check the code.

Chapter 5
Software Design

5.1 Introduction

This section describes the design of the drive's software blocks. The software description comprises these topics:

- General Flowchart
- Transmitter Software Implementation
- Receiver Software Implementation

5.2 General Flowchart

The control algorithm of a remote control transmitter and receiver are shown in Figure 5-1 and Figure 5-2, respectively. The individual detail processes are described in the software implementation sections.

5.2.1 Transmitter Flowchart

Figure 5-1 shows the overall transmitter flowchart.

After the remote control transmitter has been powered on:

- The MC68HC908LT8 registers will be initialized (such as I/O ports, timer, keyboard interrupt, and LCD driver module)
- After the register initialization phase, the keyboard interrupt is enabled and waits for any key command
- If there are no key commands, the MCU will enter stop mode for power saving
- In stop mode, all MCU modules will be off except for:
  - The sub-system clock that drives from a 32.768 kHz crystal
  - The LCD display module
  - The programmable periodic interrupt module
- If any key command is received:
  - The MCU will wakeup from stop mode
  - Decode which key command was received
  - Update the control frame data accordingly
  - The control frame will be transmitted by the IR transmitting diode
- The code will then jump back to the beginning and wait for another key command.

If there are no key presses for more than 10 seconds, the transmitter will revert back to stop mode for power saving.

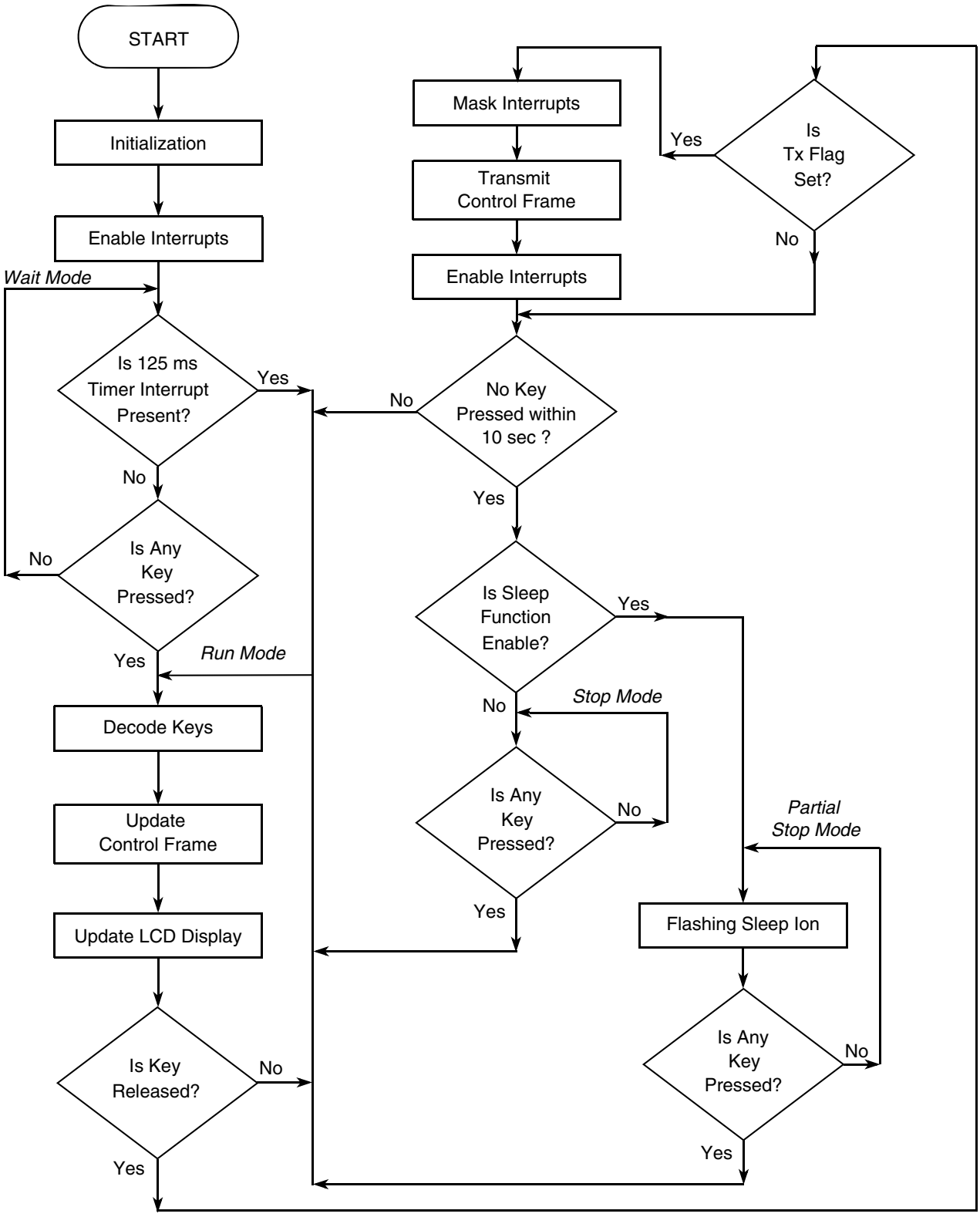


Figure 5-1. General Flowchart of Transmitter

There are two cases for standby mode, full stop mode and partial stop mode depending on the sleep function. If the sleep function is enabled, the LCD display needs to be flashing. Therefore, the LCD data register also needs to be changed by 1 second. The 1 second periodic wakeup can be implemented by the PPI module. It will wakeup the MCU periodically and change the LCD data register to flash the corresponding sleep icon(s). If the sleep function is disabled, the remote control transmitter will enter full stop mode for more power saving.

In this remote control transmitter demo, there are:

- About 20 μA for full stop mode (32.678 kHz subsystem clock, KBI enable, LCD module on)
- About 30 μA for 1 second periodic wakeup of the MCU for LCD flashing

There is a 125 ms periodic interrupt that generates from the timer. If the remote unit is in run mode, the 125 ms periodic interrupt will be generated and the control flow will be determined by a different software counter. Figure 5-1 shows one of the software counters that is used to determine the period of the key press. If no key is pressed longer that 10 seconds, the remote control transmitter unit will enter stop mode for power saving. Any key press will generate a keyboard interrupt to wakeup the MCU from stop mode and continue the operation of the remote control transmitter unit.

5.2.2 Receiver Flowchart

Figure 5-2 shows the overall receiver flowchart.

After the remote control receiver has been powered on:

- The MC68HC908LT8 MCU will initialize the internal register configuration (such as I/O ports, timer, keyboard interrupt, and LCD driver module)
- After the initialization phase, the keyboard interrupt is enabled and waits for any key command
- Since the receiver side normally does not need power saving, when no key has been pressed the receiver will enter wait mode instead of stop mode
- Next, the receiver waits for either a key press or any IR control frame signal from the transmitter
- If any key is pressed:
  - The MCU will wakeup from wait mode
  - Decode which key has been pressed
  - Update the control frame and corresponding LED and LCD display.

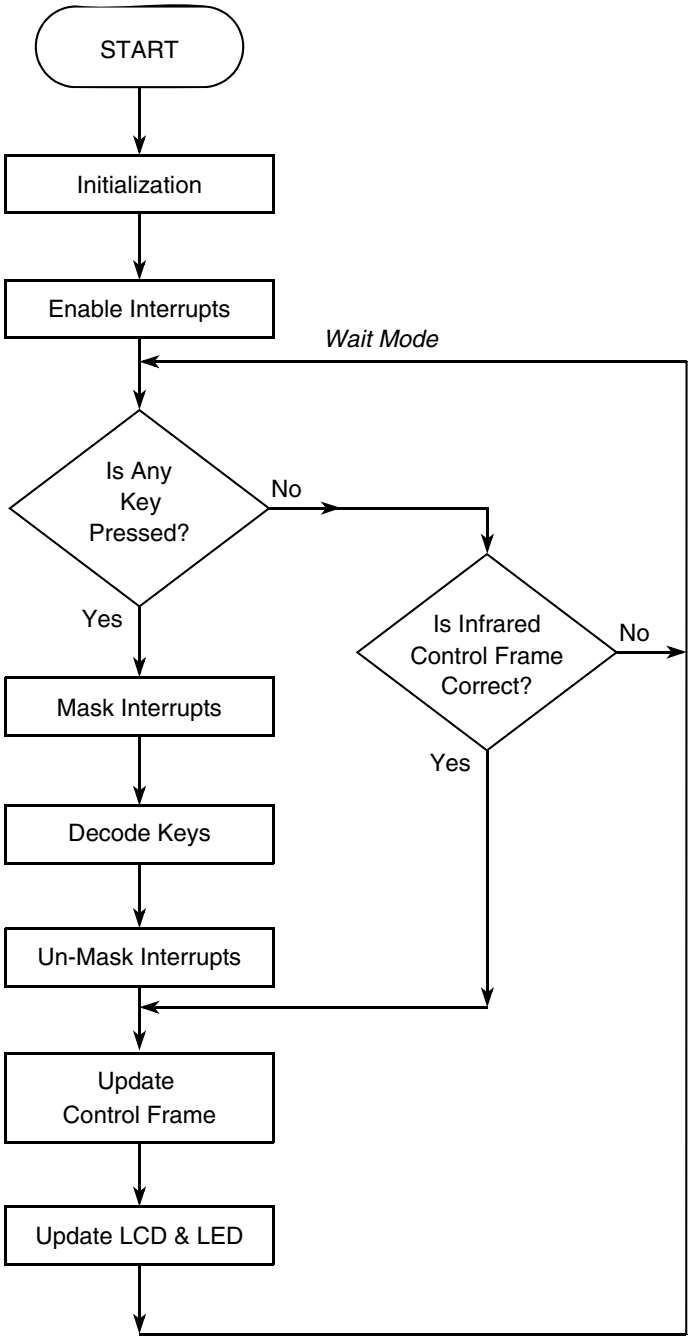


Figure 5-2. General Flowchart of Receiver

5.3 Transmitter Software Implementation

This section discusses the transmitter software implementation in detail.

5.3.1 Initialization

After transmitter power on, the MC68HC(9)08LT8 MCU will initialize the following settings:

- Initializes stack pointer
- Mask all interrupt
- Clear COP counter
- Initializes GPIO A, B, C, D, and E modules
  - GPIO A0–A2 as high outputs and A4–A6 as low outputs for the KBI matrix
  - All others of GPIO as output low
- Initializes the configuration registers
  - Disables COP and LVI
  - Disables system clock in stop mode
  - Enables subsystem clock in stop mode
  - Sets port D and E as LCD output pins
- Disables IRQ
- Sets PPI clock source to 32.768 kHz
- Enables the KBI pins and interrupts in ports A0–A3
- Initializes the LCD module
  - Sets bias voltage to 0.65% of VLCD
  - Sets bias resistors to 146 Kohm
  - Enables the LCD frame rate to 32 Hz
  - Sets 1/4 duty cycle
- Initializes the timer1 module
  - Sets clock source to bus frequency / 8
  - Sets timer overflow period to 125 ms
- Initializes the timer2 module
  - Sets clock source to bus frequency
  - Sets 38 kHz 30% duty cycle PWM in timer2
  - Sets channel 2 as a PWM output
- Initializes system variables
- Initializes control frame data
- Turns on all LCD ions for 3 seconds
- Unmasks the interrupt

After initialization, the timer1 interrupt and the 125 ms timer overflow interrupt will be enabled. They will provide the timing for the keyboard scan, LCD data update, and control frame data transmission.

5.3.2 Key Decoding

The basic configuration and general description of the transmitter keyboard scan was discussed in 4.2.2.1 Transmitter Unit. This section will focus on the software implementation of the keyboard scan.

In Figure 5-1, the key decoding subroutine is shown as “Decode Keys”. The detailed operation is shown in Figure 5-3. When any key is pressed, the keyboard interrupt service routine will be serviced and some system flag will be set. Then, key debounce will filter out any noise that was generated by key pressing. After key debounce, the key will be located by matrix scanning which is the process of scanning the row first and then the column to locate which key is pressed. There are two keys (S4 “+” and S7 “–”) that have an additional feature. If one of these keys is pressed for more than 3 seconds, the transmission control frame will update rapidly until the key is released and the transmission control frame will be transmitted out after the key is released.

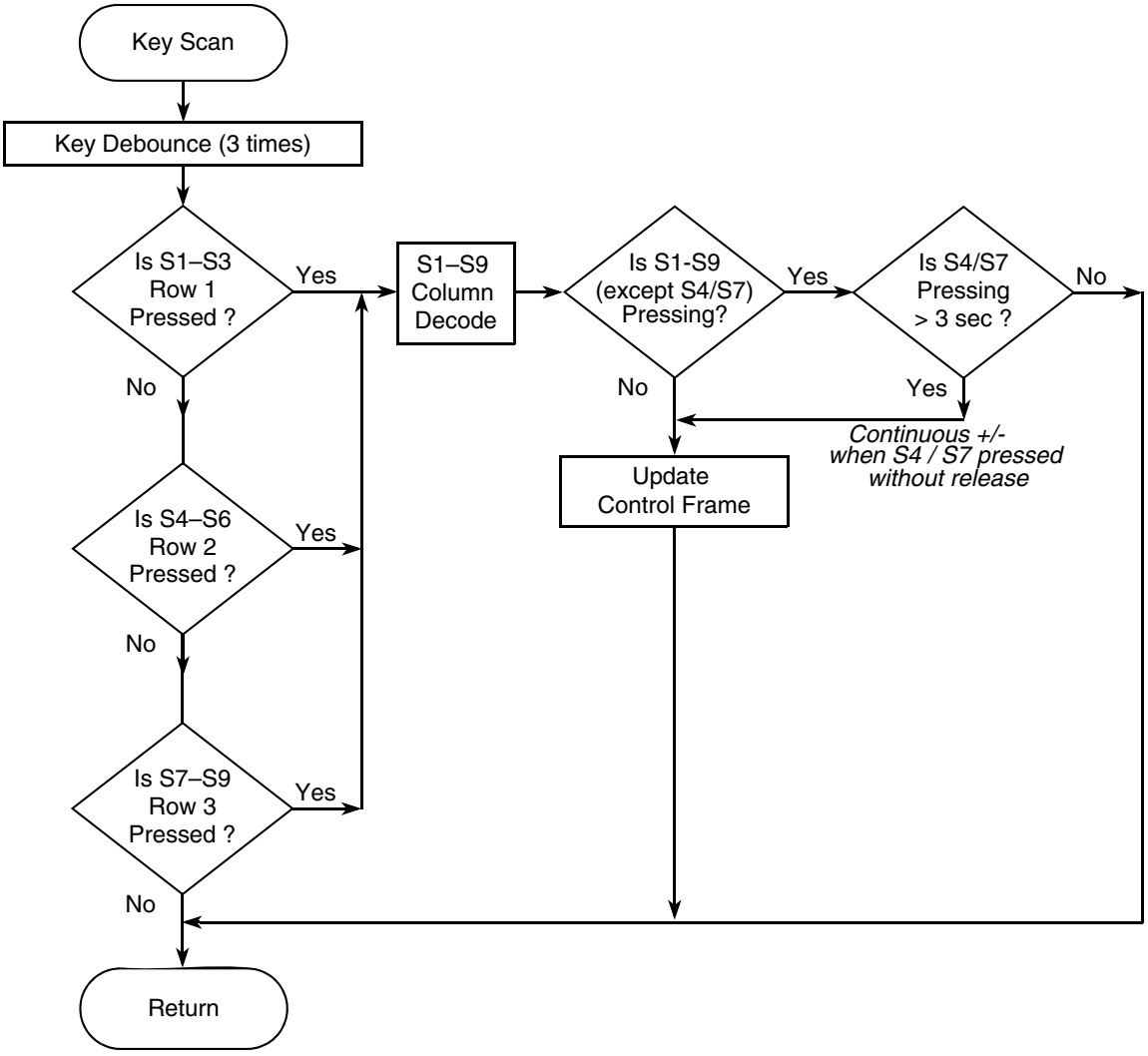


Figure 5-3. Keyboard Decoding in Transmitter

5.3.3 Transmission Control Frame Update

After locating which key is pressed, the transmission control frame will be updated based on which key is pressing. The definition of the transmission control frame is shown in Table 3-1 and the definition of the key functions are described in 4.2.2.1 Transmitter Unit.

5.3.4 LCD Display Update

The LCD display is updated every 125 ms (the timer interrupt). And, the icon on or off (in the LCD display) is dependent on the transmission control frame. After the transmission control frame is updated, the LCD display routine will decode the frame and then change the icon(s) of the display accordingly. The definition of the display is discussed in 4.3 Transmitter LCD and LED Displays.

5.4 Receiver Software Implementation

The general software flow diagram for the remote control receiver is discussed in 5.2 General Flowchart. This section will focus on a detailed description of the transmitter software implementation.

5.4.1 Initialization

After receiver power on, the MC68HC(9)08LT8 MCU will initialize the following settings:

- Initializes stack pointer
- Masks all interrupts
- Clear COP counter
- Initializes GPIO A, B, C, D, and E modules
  - GPIO A0–A1 as outputs for KBI
  - GPIO B2–B3 as outputs for the LED drive
  - All other GPIO as output low
- Initializes the configuration registers
  - Disables COP and LVI
  - Disables system clock in stop mode
  - Enables subsystem clock in stop mode
  - Sets port D and E as LCD output pins
- Disables IRQ
- Sets PPI clock source to 32.768 kHz
- Enables the KBI pins and interrupts in port A0–A1
  - Initializes the LCD module
  - Sets bias voltage to 0.65% of VLCD
  - Sets bias resistors to 146 Kohm
  - Enables LCD frame rate to 32 Hz
  - Sets 1/4 duty cycle
- Initializes system variables
- Initializes control frame data
- Turns on all LCD ions for 3 seconds
- Initializes the timer2 module
  - Sets channel 2 as an input capture
- Unmasks the interrupt

After initialization, the timer2 interrupt will be enabled and any IR control frame will cause an interrupt and the corresponding action in the receiver. On the other hand, if the KBI (PTA0–PTA1) is also enabled, any key press will cause an interrupt and the corresponding action in the receiver.

5.4.2 Key Decoding

The basic configuration and general description of the receiver keyboard decoding was discussed in 4.2.2.2 Receiver Unit. In this section, focus will be on the software implementation of the keyboard decoding.

In Figure 5-2, the key decoding subroutine is shown as “Decode Keys”. In Figure 5-4, the keyboard interrupt is used for key decoding. Similarly to transmitter key decoding, when any key is pressed it will generate a keyboard interrupt. In the case of the receiver, only two keys need to be decoded. So, there is no need to scan the key location by row and column. All that is required to poll either the S1 or S2 key to determine which key is pressed. The corresponding control frame will be updated. And, the LCD and LED display will be updated accordingly.

5.4.3 Transmission Control Frame Update

Either a key being pressed or receipt of an IR transmission control frame signal from the transmitter will cause the transmission control frame to be updated accordingly. The definition of the transmission control frame is shown in Table 3-1 And the definition of the key functions are described in 4.2.2.2 Receiver Unit.

5.4.4 LCD and LED Display Update

The LCD display is the same in both the transmitter and receiver. Since there is no flashing feature in the receiver, it does not need to update the LCD display periodically. And, the receiver requires the addition of two LEDs to indicate power on and off. Refer to 4.3 Transmitter LCD and LED Displays for more detailed information regarding the LCD and LED displays.

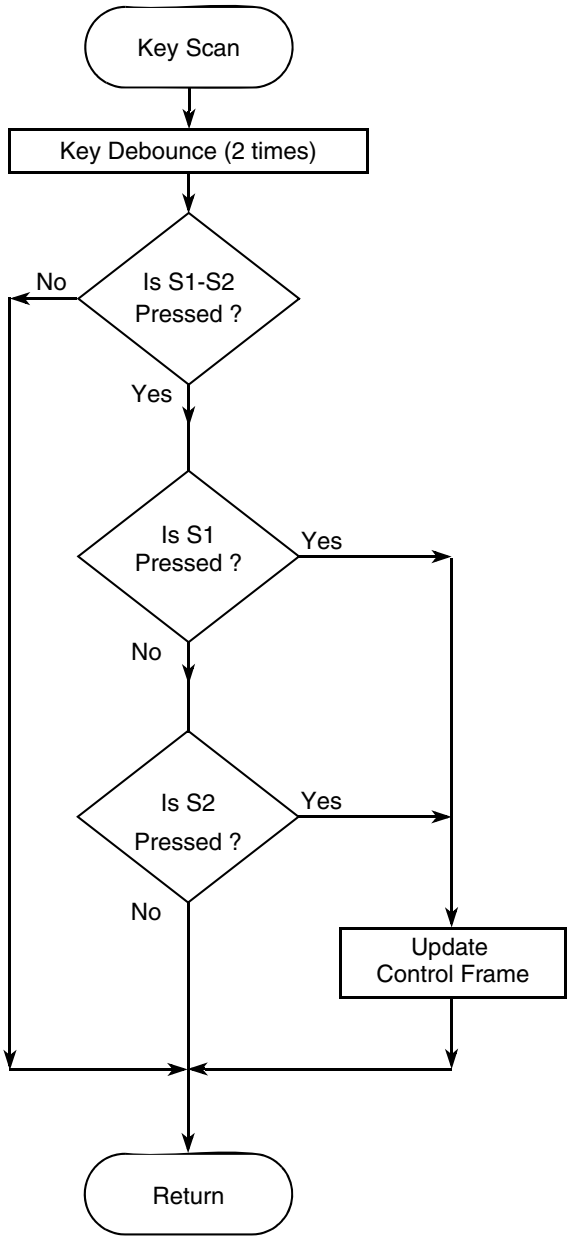


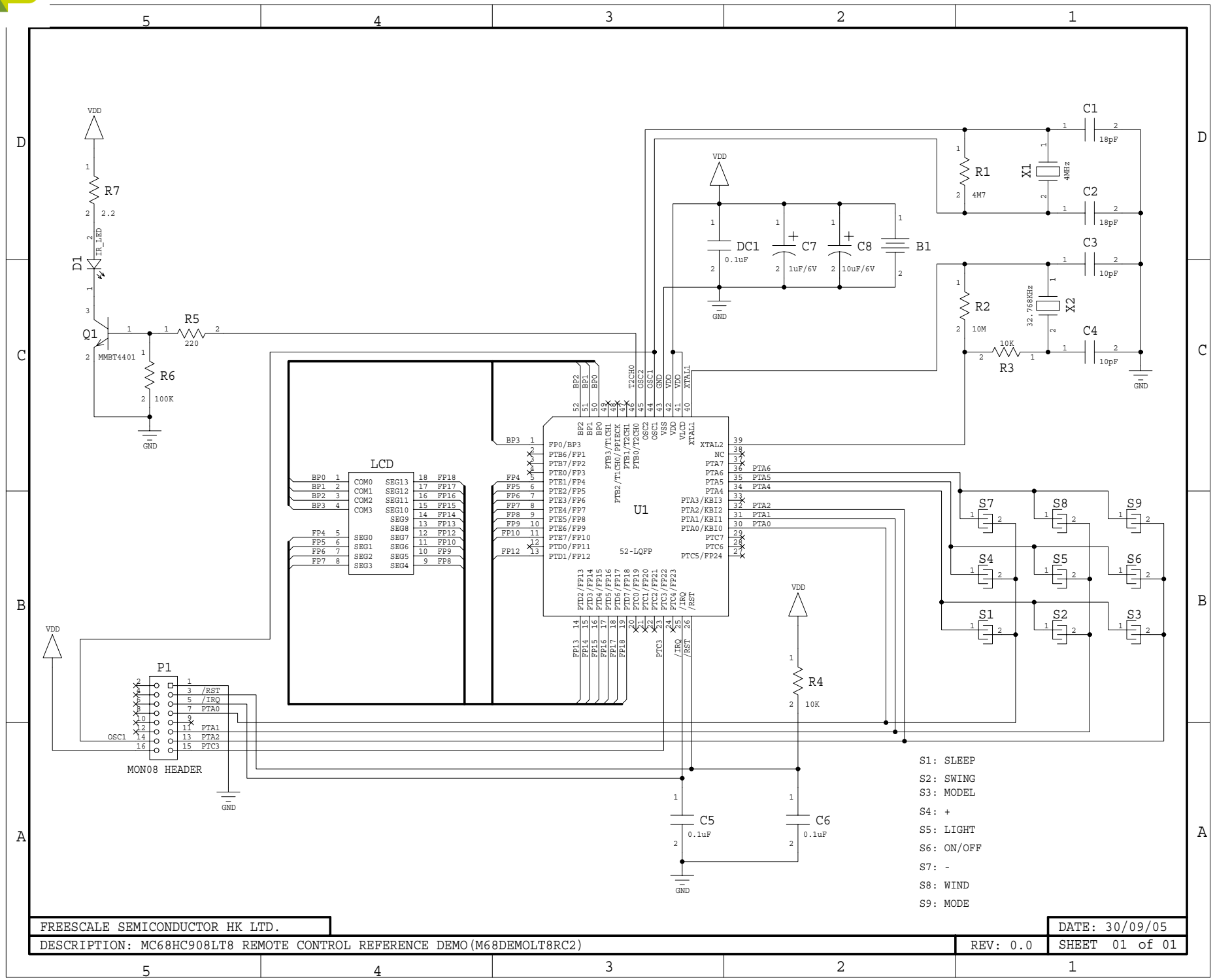
Figure 5-4. Keyboard Decoding in Receiver

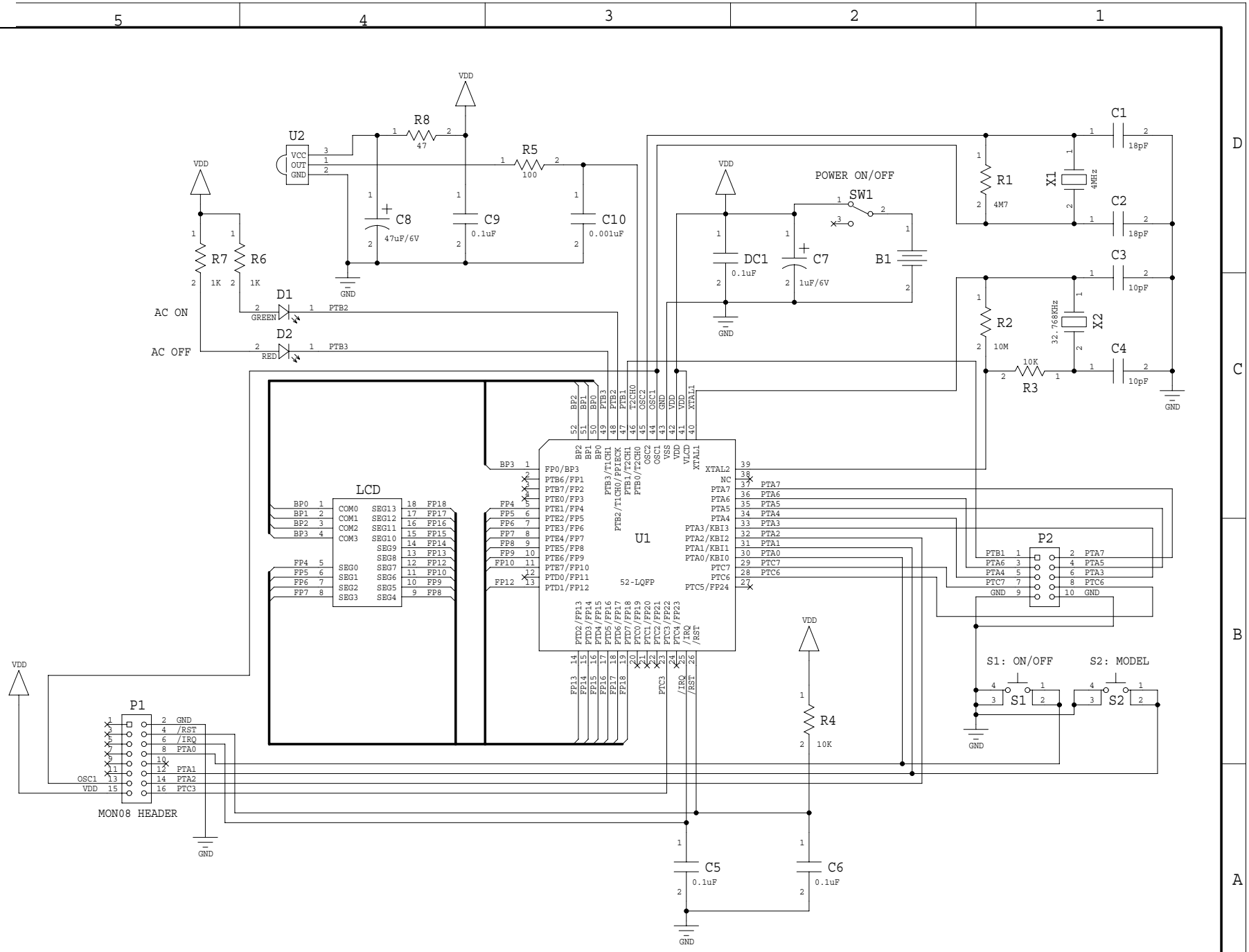
Appendix A
Schematic

A.1 Introduction

Schematics for the reference design are shown on the following pages:

MC68HC908LT8 Remote Control Transmitter Reference Demo (M68DEMOLT8RC2)
MC68HC908LT8 Remote Control Receiver Reference Demo (M68DEMO08LT8RX)


VDD
R7
2.2
D1
IR_LED
Q1
MMBT4401
R5
220
R6
100K
GND
C1
18pF
R1
4M7
X1
4MHz
C2
18pF
C3
10pF
R2
10M
X2
32.768KHz
C4
10pF
R3
10K
DC1
0.1uF
C7
1uF/6V
C8
10uF/6V
B1
LCD
COM0
COM1
COM2
COM3
SEG0
SEG1
SEG2
SEG3
SEG4
SEG5
SEG6
SEG7
SEG8
SEG9
SEG10
SEG11
SEG12
SEG13
U1
52-LQFP
XTAL2
XTAL1
NC
VDD
VSS
VLCD
OSC1
OSC2
T2CH0
T2CH1
/IRQ
/RST
P1
MON08 HEADER
R4
10K
C5
0.1uF
C6
0.1uF
S1
S2
S3
S4
S5
S6
S7
S8
S9
S1: SLEEP
S2: SWING
S3: MODEL
S4: +
S5: LIGHT
S6: ON/OFF
S7: -
S8: WIND
S9: MODE
FREESCALE SEMICONDUCTOR HK LTD.
DESCRIPTION: MC68HC908LT8 REMOTE CONTROL REFERENCE DEMO(M68DEMOLT8RC2)
REV: 0.0
DATE: 30/09/05
SHEET 01 of 01



FREESCALE SEMICONDUCTOR HK LTD.			DATE: 15/12/05
DESCRIPTION: MC68HC908LT8 REMOTE CONTROL RECEIVER REFERENCE DEMO(M68DEMO08LT8RX)		REV: 0.1	SHEET 01 of 01

Appendix B
Program Listing

B.1 Transmitter Listing

```
; ------------------------------------------------------------------------------ *
; Freescale Semiconductor (H.K.) Ltd.                                            *
; 8/16 bit MCU - Application                                                     *
;                                                                                *
; FileName      :  Main.asm                                                      *
; Title         :  08LT8 Remote Control Reference Demo code (Freescale)          *
; MCU           :  PC68HC908LT8CFB (44-LQFP) 1st Silicon (Mask Set 0M48C) only   *
; Assembler     :  Metrowerks CodeWarrior HC(S)08  (v3.1)                        *
; Include File  :  908LT8v0r0.inc (for LT8 1st silicon only)                     *
; Author        :  T.C. Lun                                                      *
;                                                                                *
;                  DD/MM/YY     Rev.    Modified comments                        *
; History       :  15/12/05     0.0     Initial release                          *
;                                                                                *
; Introduction  : The H/W setting are show as below:                             *
;    PTA[2:0]   : KBI of Key Matrix, pullup if enable                            *
;    PTA[6:4]   : o/p port of Key Scan (3x3 Matrix)                              *
;    PTA3 & 7   : NC set o/p low                                                 *
;    PTB0/T2CH0 : Tx Diode (38.46KHz 0.308% ON Duty Cycle)                       *
;    BP0-3      : Connect to LCD Panel Pin4-1                                    *
;    FP4-10     : Connect to LCD Panel Pin5-11                                   *
;    FP12-18    : Connect to LCD Panel Pin12-18                                  *
;    FP3,11 & FP19-22 : NC o/p                                                   *
;    PTB1-3     : NC set o/p low                                                 *
;    RST        : 10K pullup + 0.1uF to GND                                      *
;    IRQ        : NC connect 0.1uF to GND                                        *
;                 OSC1 & 2 : 4M7 + 4MHz + 18pF x2 OR (1M + 4MHz Resonator)      *
;    XTAL1 & 2  : 10M + 10K + 32.768KHz + 10pF x2                                *
;                 Vdd & Vss : 0.01uF // 10uF                                     *
;    Unused pin : PTB6-7 & PTC4-7 (need to set as o/p low) available in 52-LQFP  *
;                                                                                *
;  Special arrangment : Add 100uF near to Vdd of Tx diode                        *
;    : Add Mon08 interface for programming/ICD                                   *
;                                                                                *
; MON08 Interface:                                                               *
;  PIN# Net Name    | PIN# Net Name                                              *
;  1  NC            |  2   GND                                                   *
;  3  NC            |  4   /RST                                                  *
;  5  NC            |  6   /IRQ(+100 ohm)                                        *
;  7  NC            |  8   PTA0                                                  *
;  9  NC            |  10  NC                                                    *
;  11 NC            |  12  PTA1                                                  *
;  13 OSC1(+22 ohm) |  14  PTA2                                                  *
;  15 Vdd           |  16  PTC3                                                  *
```

```
;                                                                              *
;                                                                              *
; Remark : This code is only for 1st 908LT8 silicon                            *
;          For new silicon, it need to change the PPI interrupt vector         *
;          and its register & handling !!!!                                    *
;                                                                              *
;                                                                              *
; ---------------------------------------------------------------------------- *
; Disclaimer of All Warranties & Liabilities :                                 *
;  This Program is a freeware to demonstrate the operation of HC08 micro-      *
;  controller. In no event will Motorola be liable for any damages, or any     *
;  incidental or consequential damages arising out of the use of or            *
;  inability to use this program. User agrees that Motorola does not make      *
;  any warranties of any kind that the program does not or will not            *
;  infringe any copyright, patent, trade secret or other intellectual          *
;  property right of any third party in any country.                           *
; ---------------------------------------------------------------------------- *

            XDEF Entry, IRQ_ISR, main, LVI_ISR, T1M0_ISR, T2M0_ISR, T1M1_ISR,
            T2M1_ISR, T1OF_ISR, T2OF_ISR, KBI_ISR, DMY_ISR

            Include '908LT8v0r0.inc'
;*******************************************************************************
;* Title: 908LV8.inc   (c) Freescale Semiconductor, Inc. 2004 All rights reserved
;*******************************************************************************
;* Author: T.C. Lun - Freescale TSPG
;*
;* Description: Register and bit name definitions for MC68HC908LT8
;*
;* Documentation:  MC68HC908LT8/D
;*
;* Include Files: none
;*
;* Assembler:  Metrowerks Code Warrior 3.0
;*             or P&E Microcomputer Systems - CASM08Z (v. 3.16)
;*
;* Revision History:
;* Rev #     Date      Who     Comments
;* -----  -----------  ------  ----------------------------------------------
;*  0.0    10/14/04    TC.Lun   (Rev.0.3 Preliminary Draft is used)
;*
;*******************************************************************************
;****  Memory Map and Interrupt Vectors  ***************************************
;*
RamStart:   equ   $0080       ;start of RAM
RamLast:    equ   $00FF       ;last RAM location
RomStart:   equ   $DE00       ;start of Flash
RomLast:    equ   $FFFF       ;last Flash location
MonStart:   equ   $0B97       ;start of monitor ROM

Vkbd:       equ   $FFDC       ;keyboard vector
Vtim2ov:    equ   $FFEA       ;timer 2 overflow
Vtim2ch1:   equ   $FFEC       ;timer 2 channel 1 vector
Vtim2ch0:   equ   $FFEE       ;timer 2 channel 0 vector
```

```
Vtimov:     equ   $FFF0       ;timer 1 overflow vector
Vtimch1:    equ   $FFF2       ;timer 1 channel 1 vector
Vtimch0:    equ   $FFF4       ;timer 1 channel 0 vector
Vlvi:       equ   $FFF8       ;LVI interrupt vector
Virq:       equ   $FFFA       ;IRQ vector
Vswi:       equ   $FFFC       ;SWI vector
Vreset:     equ   $FFFE       ;reset vector

;****  Input/Output (I/O) Ports  ************************************************
;*
PTA:        equ   $00         ;port A data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTA7:       equ   7           ;port A data bit 7
PTA6:       equ   6           ;port A data bit 6
PTA5:       equ   5           ;port A data bit 5
PTA4:       equ   4           ;port A data bit 4
PTA3:       equ   3           ;port A data bit 3
PTA2:       equ   2           ;port A data bit 2
PTA1:       equ   1           ;port A data bit 1
PTA0:       equ   0           ;port A data bit 0
; bit position masks
mPTA7:      equ   %10000000   ;port A data bit 7
mPTA6:      equ   %01000000   ;port A data bit 6
mPTA5:      equ   %00100000   ;port A data bit 5
mPTA4:      equ   %00010000   ;port A data bit 4
mPTA3:      equ   %00001000   ;port A data bit 3
mPTA2:      equ   %00000100   ;port A data bit 2
mPTA1:      equ   %00000010   ;port A data bit 1
mPTA0:      equ   %00000001   ;port A data bit 0

PTB:        equ   $01         ;port B data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTB7:       equ   7           ;port B data bit 7
PTB6:       equ   6           ;port B data bit 6
PTB3:       equ   3           ;port B data bit 3
PTB2:       equ   2           ;port B data bit 2
PTB1:       equ   1           ;port B data bit 1
PTB0:       equ   0           ;port B data bit 0
; bit position masks
mPTB7:      equ   %10000000   ;port B data bit 7
mPTB6:      equ   %01000000   ;port B data bit 6
mPTB3:      equ   %00001000   ;port B data bit 3
mPTB2:      equ   %00000100   ;port B data bit 2
mPTB1:      equ   %00000010   ;port B data bit 1
mPTB0:      equ   %00000001   ;port B data bit 0

PTC:        equ   $02         ;port C data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTC7:       equ   7           ;port C data bit 7
PTC6:       equ   6           ;port C data bit 6
PTC5:       equ   5           ;port C data bit 5
PTC4:       equ   4           ;port C data bit 4
PTC3:       equ   3           ;port C data bit 3
PTC2:       equ   2           ;port C data bit 2
```

```
PTC1:       equ   1           ;port C data bit 1
PTC0:       equ   0           ;port C data bit 0
; bit position masks
mPTC7:      equ   %10000000   ;port C data bit 7
mPTC6:      equ   %01000000   ;port C data bit 6
mPTC5:      equ   %00100000   ;port C data bit 5
mPTC4:      equ   %00010000   ;port C data bit 4
mPTC3:      equ   %00001000   ;port C data bit 3
mPTC2:      equ   %00000100   ;port C data bit 2
mPTC1:      equ   %00000010   ;port C data bit 1
mPTC0:      equ   %00000001   ;port C data bit 0

PTD:        equ   $03         ;port D data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTD7:       equ   7           ;port D data bit 7
PTD6:       equ   6           ;port D data bit 6
PTD5:       equ   5           ;port D data bit 5
PTD4:       equ   4           ;port D data bit 4
PTD3:       equ   3           ;port D data bit 3
PTD2:       equ   2           ;port D data bit 2
PTD1:       equ   1           ;port D data bit 1
PTD0:       equ   0           ;port D data bit 0
; bit position masks
mPTD7:      equ   %10000000   ;port D data bit 7
mPTD6:      equ   %01000000   ;port D data bit 6
mPTD5:      equ   %00100000   ;port D data bit 5
mPTD4:      equ   %00010000   ;port D data bit 4
mPTD3:      equ   %00001000   ;port D data bit 3
mPTD2:      equ   %00000100   ;port D data bit 2
mPTD1:      equ   %00000010   ;port D data bit 1
mPTD0:      equ   %00000001   ;port D data bit 0

DDRA:       equ   $04         ;port A data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRA7:      equ   7           ;port A data direction bit 7
DDRA6:      equ   6           ;port A data direction bit 6
DDRA5:      equ   5           ;port A data direction bit 5
DDRA4:      equ   4           ;port A data direction bit 4
DDRA3:      equ   3           ;port A data direction bit 3
DDRA2:      equ   2           ;port A data direction bit 2
DDRA1:      equ   1           ;port A data direction bit 1
DDRA0:      equ   0           ;port A data direction bit 0
; bit position masks
mDDRA7:     equ   %10000000   ;port A data direction bit 7
mDDRA6:     equ   %01000000   ;port A data direction bit 6
mDDRA5:     equ   %00100000   ;port A data direction bit 5
mDDRA4:     equ   %00010000   ;port A data direction bit 4
mDDRA3:     equ   %00001000   ;port A data direction bit 3
mDDRA2:     equ   %00000100   ;port A data direction bit 2
mDDRA1:     equ   %00000010   ;port A data direction bit 1
mDDRA0:     equ   %00000001   ;port A data direction bit 0
```

```
DDRB:       equ   $05         ;port B data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRB7:      equ   7           ;port B data direction bit 7
DDRB6:      equ   6           ;port B data direction bit 6
DDRB3:      equ   3           ;port B data direction bit 3
DDRB2:      equ   2           ;port B data direction bit 2
DDRB1:      equ   1           ;port B data direction bit 1
DDRB0:      equ   0           ;port B data direction bit 0
; bit position masks
mDDRB7:     equ   %10000000   ;port B data direction bit 7
mDDRB6:     equ   %01000000   ;port B data direction bit 6
mDDRB3:     equ   %00001000   ;port B data direction bit 3
mDDRB2:     equ   %00000100   ;port B data direction bit 2
mDDRB1:     equ   %00000010   ;port B data direction bit 1
mDDRB0:     equ   %00000001   ;port B data direction bit 0

DDRC:       equ   $06         ;port C data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRC7:      equ   7           ;port C data direction bit 7
DDRC6:      equ   6           ;port C data direction bit 6
DDRC5:      equ   5           ;port C data direction bit 5
DDRC4:      equ   4           ;port C data direction bit 4
DDRC3:      equ   3           ;port C data direction bit 3
DDRC2:      equ   2           ;port C data direction bit 2
DDRC1:      equ   1           ;port C data direction bit 1
DDRC0:      equ   0           ;port C data direction bit 0
; bit position masks
mDDRC7:     equ   %10000000   ;port C data direction bit 7
mDDRC6:     equ   %01000000   ;port C data direction bit 6
mDDRC5:     equ   %00100000   ;port C data direction bit 5
mDDRC4:     equ   %00010000   ;port C data direction bit 4
mDDRC3:     equ   %00001000   ;port C data direction bit 3
mDDRC2:     equ   %00000100   ;port C data direction bit 2
mDDRC1:     equ   %00000010   ;port C data direction bit 1
mDDRC0:     equ   %00000001   ;port C data direction bit 0

DDRD:       equ   $07         ;port D data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRD7:      equ   7           ;port D data direction bit 7
DDRD6:      equ   6           ;port D data direction bit 6
DDRD5:      equ   5           ;port D data direction bit 5
DDRD4:      equ   4           ;port D data direction bit 4
DDRD3:      equ   3           ;port D data direction bit 3
DDRD2:      equ   2           ;port D data direction bit 2
DDRD1:      equ   1           ;port D data direction bit 1
DDRD0:      equ   0           ;port D data direction bit 0
; bit position masks
mDDRD7:     equ   %10000000   ;port D data direction bit 7
mDDRD6:     equ   %01000000   ;port D data direction bit 6
mDDRD5:     equ   %00100000   ;port D data direction bit 5
mDDRD4:     equ   %00010000   ;port D data direction bit 4
mDDRD3:     equ   %00001000   ;port D data direction bit 3
mDDRD2:     equ   %00000100   ;port D data direction bit 2
mDDRD1:     equ   %00000010   ;port D data direction bit 1
```

```
mDDRD0:     equ   %00000001   ;port D data direction bit 0

DDRE:       equ   $08         ;port E data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRE7:      equ   7           ;port E data direction bit 7
DDRE6:      equ   6           ;port E data direction bit 6
DDRE5:      equ   5           ;port E data direction bit 5
DDRE4:      equ   4           ;port E data direction bit 4
DDRE3:      equ   3           ;port E data direction bit 3
DDRE2:      equ   2           ;port E data direction bit 2
DDRE1:      equ   1           ;port E data direction bit 1
DDRE0:      equ   0           ;port E data direction bit 0
; bit position masks
mDDRE7:     equ   %10000000   ;port E data direction bit 7
mDDRE6:     equ   %01000000   ;port E data direction bit 6
mDDRE5:     equ   %00100000   ;port E data direction bit 5
mDDRE4:     equ   %00010000   ;port E data direction bit 4
mDDRE3:     equ   %00001000   ;port E data direction bit 3
mDDRE2:     equ   %00000100   ;port E data direction bit 2
mDDRE1:     equ   %00000010   ;port E data direction bit 1
mDDRE0:     equ   %00000001   ;port E data direction bit 0

PTE:        equ   $09         ;port E data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTE7:       equ   7           ;port E data bit 7
PTE6:       equ   6           ;port E data bit 6
PTE5:       equ   5           ;port E data bit 5
PTE4:       equ   4           ;port E data bit 4
PTE3:       equ   3           ;port E data bit 3
PTE2:       equ   2           ;port E data bit 2
PTE1:       equ   1           ;port E data bit 1
PTE0:       equ   0           ;port E data bit 0
; bit position masks
mPTE7:      equ   %10000000   ;port E data bit 7
mPTE6:      equ   %01000000   ;port E data bit 6
mPTE5:      equ   %00100000   ;port E data bit 5
mPTE4:      equ   %00010000   ;port E data bit 4
mPTE3:      equ   %00001000   ;port E data bit 3
mPTE2:      equ   %00000100   ;port E data bit 2
mPTE1:      equ   %00000010   ;port E data bit 1
mPTE0:      equ   %00000001   ;port E data bit 0

HDB:        equ   $0C         ;port B high current drive control
                              ;register
; bit number for use in BCLR, BSET, BRCLR, BRSET
PPI1L:      equ   6           ;PPI1 interrupt request level
HDB3:       equ   3           ;port B3 high current drive enable
HDB2:       equ   2           ;port B2 high current drive enable
PPI1CLKS1:  equ   1           ;PPI1 clock select 1
PPI1CLKS0:  equ   0           ;PPI1 clock select 0

mPPI1L:     equ   %01000000   ;PPI1 interrupt request level
mHDB3:      equ   %00001000   ;port B3 high current drive enable
mHDB2:      equ   %00000100   ;port B2 high current drive enable
```

```
mPPI1CLKS1: equ   %00000010   ;PPI1 clock select 1
mPPI1CLKS0: equ   %00000001   ;PPI1 clock select 0

;****  Keyboard Interrupt Module (KBI)  ****************************************
;*
KBSCR:      equ   $1B         ;keyboard status and control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
KEYF:       equ   3           ;keyboard flag
ACKK:       equ   2           ;keyboard acknowledge
IMASKK:     equ   1           ;keyboard interrupt mask
MODEK:      equ   0           ;keyboard triggering sesitivity
; bit position masks
mKEYF:      equ   %00001000   ;keyboard flag
mACKK:      equ   %00000100   ;keyboard acknowledge
mIMASKK:    equ   %00000010   ;keyboard interrupt mask
mMODEK:     equ   %00000001   ;keyboard triggering sesitivity

KBIER:      equ   $1C         ;keyboard interrupt enable register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
PPI1IE2:    equ   6           ;PPI1 interrupt enable and frequency
                              ;select bit 2
PPI1IE1:    equ   5           ;PPI1 interrupt enable and frequency
                              ;select bit 1
PPI1IE0:    equ   4           ;PPI1 interrupt enable and frequency
                              ;select bit 0
KBIE3:      equ   3           ;port A keyboard interrupt enable bit 3
KBIE2:      equ   2           ;port A keyboard interrupt enable bit 2
KBIE1:      equ   1           ;port A keyboard interrupt enable bit 1
KBIE0:      equ   0           ;port A keyboard interrupt enable bit 0
; bit position masks
mPPI1IE2:   equ   %01000000   ;PPI1 interrupt enable and frequency
                              ;select bit 2
mPPI1IE1:   equ   %00100000   ;PPI1 interrupt enable and frequency
                              ;select bit 1
mPPI1IE0:   equ   %00010000   ;PPI1 interrupt enable and frequency
                              ;select bit 0
mKBIE3:     equ   %00001000   ;port A keyboard interrupt enable bit 3
mKBIE2:     equ   %00000100   ;port A keyboard interrupt enable bit 2
mKBIE1:     equ   %00000010   ;port A keyboard interrupt enable bit 1
mKBIE0:     equ   %00000001   ;port A keyboard interrupt enable bit 0

;**** Configuration Registers 2 (CONFIG2) ************************************
;*
CONFIG2:    equ   $1D         ;configuration register 2
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
STOP_XCLKEN: equ   7          ;stop osc1 & osc2 crystal clock enable
STOP_XTALEN: equ   6          ;stop xtal1 & xtal2 crystal clock enable
PEE:        equ   5           ;port E LCD or GPIO select
PDE:        equ   4           ;port D LCD or GPIO select
PCEH:       equ   3           ;port C higher nibble lcd or GPIO select
PCEL:       equ   2           ;port C lower nibble lcd or GPIO select
LVISEL1:    equ   1           ;LVI level select bit 1
LVISEL0:    equ   0           ;LVI level select bit 0
```

```
; bit position masks
mSTOP_XCLKEN: equ   %10000000 ;stop osc1 & osc2 crystal clock enable
mSTOP_XTALEN: equ   %01000000 ;stop xtal1 & xtal2 crystal clock enable
mPEE:       equ   %00100000   ;port E LCD or GPIO select
mPDE:       equ   %00010000   ;port D LCD or GPIO select
mPCEH:      equ   %00001000   ;port C higher nibble lcd or GPIO select
mPCEL:      equ   %00000100   ;port C lower nibble lcd or GPIO select
mLVISEL1:   equ   %00000010   ;LVI level select bit 1
mLVISEL0:   equ   %00000001   ;LVI level select bit 0

;****  External Interrupt (IRQ)  ***********************************************
;*
INTSCR:     equ   $1E         ;IRQ status and control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
IRQF:       equ   3           ;IRQ flag
ACK:        equ   2           ;IRQ interrupt request acknowledge
IMASK:      equ   1           ;IRQ interrupt mask
MODE:       equ   0           ;IRQ edge/level select
; bit position masks
mIRQF:      equ   %00001000   ;IRQ flag
mACK:       equ   %00000100   ;IRQ interrupt request acknowledge
mIMASK:     equ   %00000010   ;IRQ interrupt mask
mMODE:      equ   %00000001   ;IRQ edge/level select

;**** Configuration Register 1 (CONFIG1) ***************************************
;*
CONFIG1:    equ   $1F         ;configuration register 1
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
COPRS:      equ   7           ;COP reset period selection
LVISTOP:    equ   6           ;LVI enable in stop mode
LVIRSTD:    equ   5           ;LVI reset disable
LVIPWRD:    equ   4           ;LVI power disable
SSREC:      equ   2           ;short stop recovery
STOP:       equ   1           ;STOP instruction enable
COPD:       equ   0           ;COP disable
; bit position masks
mCOPRS:     equ   %10000000   ;COP reset period selection
mLVISTOP:   equ   %01000000   ;LVI enable in stop mode
mLVIRSTD:   equ   %00100000   ;LVI reset disable
mLVIPWRD:   equ   %00010000   ;LVI power disable
mSSREC:     equ   %00000100   ;short stop recovery
mSTOP:      equ   %00000010   ;STOP instruction enable
mCOPD:      equ   %00000001   ;COP disable

;****  Timer Interface module 1 (TIM1)  ****************************************
;*
T1SC:       equ   $20         ;timer 1 status and control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
TOF:        equ   7           ;TIM overflow flag
TOIE:       equ   6           ;TIM overflow interrupt enable
TSTOP:      equ   5           ;TIM stop bit
TRST:       equ   4           ;TIM reset bit
PS2:        equ   2           ;prescaler select bit 2
PS1:        equ   1           ;prescaler select bit 1
```

```
PS0:        equ   0           ;prescaler select bit 0
; bit position masks
mTOF:       equ   %10000000   ;TIM overflow flag
mTOIE:      equ   %01000000   ;TIM overflow interrupt enable
mTSTOP:     equ   %00100000   ;TIM stop bit
mTRST:      equ   %00010000   ;TIM reset bit
mPS2:       equ   %00000100   ;prescaler select bit 2
mPS1:       equ   %00000010   ;prescaler select bit 1
mPS0:       equ   %00000001   ;prescaler select bit 0

T1SC0:      equ   $25         ;timer 1 channel 0 status and control
                              ;register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
CH0F:       equ   7           ;channel 0 flag
CH0IE:      equ   6           ;channel 0 interrupt enable
MS0B:       equ   5           ;mode select bit B
MS0A:       equ   4           ;mode select bit A
ELS0B:      equ   3           ;edge/level select bit B
ELS0A:      equ   2           ;edge/level select bit A
TOV0        equ   1           ;toggle on overflow
CH0MAX      equ   0           ;channel 0 maximum duty cycle
; bit position masks
mCH0F:      equ   %10000000   ;channel 0 flag
mCH0IE:     equ   %01000000   ;channel 0 interrupt enable
mMS0B:      equ   %00100000   ;mode select bit B
mMS0A:      equ   %00010000   ;mode select bit A
mELS0B:     equ   %00001000   ;edge/level select bit B
mELS0A:     equ   %00000100   ;edge/level select bit A
mTOV0       equ   %00000010   ;toggle on overflow
mCH0MAX     equ   %00000001   ;channel 0 maximum duty cycle

T1SC1:      equ   $28         ;timer 1 channel 1 status and control
                              ;register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
CH1F:       equ   7           ;channel 1 flag
CH1IE:      equ   6           ;channel 1 interrupt enable
MS1A:       equ   4           ;mode select bit A
ELS1B:      equ   3           ;edge/level select bit B
ELS1A:      equ   2           ;edge/level select bit A
TOV1        equ   1           ;toggle on overflow
CH1MAX      equ   0           ;channel 1 maximum duty cycle
; bit position masks
mCH1F:      equ   %10000000   ;channel 1 flag
mCH1IE:     equ   %01000000   ;channel 1 interrupt enable
mMS1A:      equ   %00010000   ;mode select bit A
mELS1B:     equ   %00001000   ;edge/level select bit B
mELS1A:     equ   %00000100   ;edge/level select bit A
mTOV1       equ   %00000010   ;toggle on overflow
mCH1MAX     equ   %00000001   ;channel 1 maximum duty cycle

T1CNTH:     equ   $21         ;timer 1 counter register high
T1CNTL:     equ   $22         ;timer 1 counter register Low
T1MODH:     equ   $23         ;timer 1 counter modulo register high
T1MODL:     equ   $24         ;timer 1 counter modulo register low
```

```
T1CH0H:     equ   $26         ;timer 1 channel 0 register high
T1CH0L:     equ   $27         ;timer 1 channel 0 register low
T1CH1H:     equ   $29         ;timer 1 channel 1 register high
T1CH1L:     equ   $2A         ;timer 1 channel 1 register low

;****  Timer Interface module 2 (TIM2)  ************************************
;*
T2SC:       equ   $2B         ;timer 2 status and control register
T2CNTH:     equ   $2C         ;timer 2 counter register high
T2CNTL:     equ   $2D         ;timer 2 counter register low
T2MODH:     equ   $2E         ;timer 2 counter modulo register high
T2MODL:     equ   $2F         ;timer 2 counter modulo register low
T2SC0:      equ   $30         ;timer 2 channel 0 status and control
                              ;register
T2CH0H:     equ   $31         ;timer 2 channel 0 register high
T2CH0L:     equ   $32         ;timer 2 channel 0 register low
T2SC1:      equ   $33         ;timer 2 channel 1 status and control
                              ;register
T2CH1H:     equ   $34         ;timer 2 channel 1 register high
T2CH1L:     equ   $35         ;timer 2 channel 1 register low

;****  LCD Driver  *********************************************************
;*
LCDCLK:     equ   $4F         ;LCD clock register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
FCCTL1:     equ   6           ;fast charge duty cycle select 1
FCCTL0:     equ   5           ;fast charge duty cycle select 0
DUTY1:      equ   4           ;duty cycle select 1
DUTY0:      equ   3           ;duty cycle select 0
LCLK2:      equ   2           ;LCD clock select bit 2
LCLK1:      equ   1           ;LCD clock select bit 1
LCLK0:      equ   0           ;LCD clock select bit 0
; bit position masks
mFCCTL1:    equ   %01000000   ;fast charge duty cycle select 1
mFCCTL0:    equ   %00100000   ;fast charge duty cycle select 0
mDUTY1:     equ   %00010000   ;duty cycle select 1
mDUTY0:     equ   %00001000   ;duty cycle select 0
mLCLK2:     equ   %00000100   ;LCD clock select bit 2
mLCLK1:     equ   %00000010   ;LCD clock select bit 1
mLCLK0:     equ   %00000001   ;LCD clock select bit 0

LCDCR:      equ   $51         ;LCD control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
LCDE:       equ   7           ;LCD enable
FC:         equ   5           ;fast charge
LC:         equ   4           ;Low current
LCCON3:     equ   3           ;LCD contrast control bit3
LCCON2:     equ   2           ;LCD contrast control bit2
LCCON1:     equ   1           ;LCD contrast control bit1
LCCON0:     equ   0           ;LCD contrast control bit0
; bit position masks
mLCDE:      equ   %10000000   ;LCD enable
mFC:        equ   %00100000   ;fast charge
mLC:        equ   %00010000   ;Low current
```

```
mLCCON3:    equ   %00001000   ;LCD contrast control bit3
mLCCON2:    equ   %00000100   ;LCD contrast control bit2
mLCCON1:    equ   %00000010   ;LCD contrast control bit1
mLCCON0:    equ   %00000001   ;LCD contrast control bit0

LDAT1:      equ   $52         ;LCD display data register 1
LDAT2:      equ   $53         ;LCD display data register 2
LDAT3:      equ   $54         ;LCD display data register 3
LDAT4:      equ   $55         ;LCD display data register 4
LDAT5:      equ   $56         ;LCD display data register 5
LDAT6:      equ   $57         ;LCD display data register 6
LDAT7:      equ   $58         ;LCD display data register 7
LDAT8:      equ   $59         ;LCD display data register 8
LDAT9:      equ   $5A         ;LCD display data register 9
LDAT10:     equ   $5B         ;LCD display data register 10
LDAT11:     equ   $5C         ;LCD display data register 11
LDAT12:     equ   $5D         ;LCD display data register 12
LDAT13:     equ   $5E         ;LCD display data register 13

;****  System Integration Module (SIM)  ****************************************
;*
SBSR:       equ   $FE00       ;SIM break status register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
SBSW        equ   1           ;SIM break stop/wait
; bit position masks
mSBSW:      equ   %00000010   ;SIM break stop/wait

SRSR:       equ   $FE01       ;SIM reset status register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
POR:        equ   7           ;power-on reset
PIN:        equ   6           ;external reset
COP:        equ   5           ;COP reset
ILOP:       equ   4           ;illegal opcode reset
ILAD:       equ   3           ;illegal address reset
MODRST:     equ   2           ;monitor mode entry module reset
LVI:        equ   1           ;LVI reset
; bit position masks
mPOR:       equ   %10000000   ;power-on reset
mPIN:       equ   %01000000   ;external reset
mCOP:       equ   %00100000   ;COP reset
mILOP:      equ   %00010000   ;illegal opcode reset
mILAD:      equ   %00001000   ;illegal address reset
mMODRST:    equ   %00000100   ;monitor mode entry module reset
mLVI:       equ   %00000010   ;LVI reset

SBFCR:      equ   $FE03       ;SIM break flag control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
BCFE:       equ   7           ;break clear flag enable
; bit position masks
mBCFE:      equ   %10000000   ;break clear flag enable

INT1:       equ   $FE04       ;interrupt status register 1
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
IF6:        equ   7           ;interrupt flag 6
```

```
IF5:        equ   6           ;interrupt flag 5
IF4:        equ   5           ;interrupt flag 4
IF3:        equ   4           ;interrupt flag 3
IF2:        equ   3           ;interrupt flag 2
IF1:        equ   2           ;interrupt flag 1

INT2:       equ   $FE05       ;interrupt status register 2
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
IF9:        equ   2           ;interrupt flag 9
IF8:        equ   1           ;interrupt flag 8
IF7:        equ   0           ;interrupt flag 7

INT3:       equ   $FE06       ;interrupt status register 3
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
IF17:       equ   2           ;interrupt flag 17
IF16:       equ   1           ;interrupt flag 16

;****  Flash Memory  ************************************************************
;*
FLCR:       equ   $FE08       ;flash control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
HVEN:       equ   3           ;high-voltage enable bit mask
MASS:       equ   2           ;mass erase control bit mask
ERASE:      equ   1           ;erase control bit mask
PGM:        equ   0           ;program control bit mask
; bit position masks
mHVEN:      equ   %00001000   ;high-voltage enable bit mask
mMASS:      equ   %00000100   ;mass erase control bit mask
mERASE:     equ   %00000010   ;erase control bit mask
mPGM:       equ   %00000001   ;program control bit mask

FLBPR:      equ   $FF7E       ;flash block protect register

;****  Break Module (BRK)  ******************************************************
;*
BRKSCR:     equ   $FE0E       ;break status and control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
BRKE:       equ   7           ;break enable
BRKA:       equ   6           ;break active
; bit position masks
mBRKE:      equ   %10000000   ;break enable
mBRKA:      equ   %01000000   ;break active

BRKH:       equ   $FE0C       ;break address register high
BRKL:       equ   $FE0D       ;break address register low

;****  Low-Voltage Inhibit (LVI)  ***********************************************
;*
LVISR:      equ   $FE0F       ;LVI status register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
LVIOUT:     equ   7           ;LVI output
LVIIE:      equ   6           ;LVI interrupt enable bit
LVIIF:      equ   5           ;LVI interrupt flag
LVIIAK:     equ   4           ;LVI interrupt request acknowledge bit
```

```
; bit position masks
mLVIOUT:    equ   %10000000   ;LVI output
mLVIIE:     equ   %01000000   ;LVI interrupt enable bit
mLVIIF:     equ   %00100000   ;LVI interrupt flag
mLVIIAK:    equ   %00010000   ;LVI interrupt request acknowledge bit
;****  Computer Operating Properly (COP)  ************************************
;*
COPCTL:     equ   $FFFF       ;COP control register

;*****************************************************************************
; Registers definition
;*****************************************************************************
; PART I: MCU Related values and regsisters
; ---------------------------------------------------------------------------- *
CONFIG1_Init     equ %10110011     ; Config 1 initial
;                    ||||||||
;                    |||||||+-- COP=disable(1)
;                    ||||||+--- Stop=enable(1)
;                    |||||+---- Recovery time=long(0)
;                    ||||+----- NIL=(0)
;                    |||+------ LVI power=on(0)
;                    ||+------- LVI reset=disable(1)
;                    |+-------- LVI in stop=disable(0)
;                    +--------- COP=long(0)
;
CONFIG2_Init     equ %01111101     ; Config 2 initial
;                    ||||||||
;                    |||||||+-- LVI trip point[0] [1:0]= 1:0 = 5.0V
;                    ||||||+--- LVI trip point[1] [1:0]= 0:1 = 3.0V
;                    |||||+---- PTC0-3 = LCD(1)
;                    ||||+----- PTC4-7 = LCD(1)
;                    |||+------ PTD = LCD(1) Need set PTD=$FF (1st 908LT8)
;                    ||+------- PTE = LCD(1) Need set PTE=$FF (1st 908LT8)
;                    |+-------- XTAL enable in stop(1)
;                    +--------- OSC enable in stop(1)
;
LVISR_Init       equ %01010000     ; LVISR initial
;                    ||||||||
;                    ||||++++-- NIL=0 (bit 3-0)
;                    |||+------ LVI interrupt acknowledge (1=clear LVI int flag)
;                    ||+------- LVI interrupt flag (1=pending)
;                    |+-------- LVI interrupt enable (1=enable)
;                    +--------- LVI o/p flag(=1 if Vdd fall below LVI trip pt)
;
INTSCR_Init      equ %00000110     ; INTSCR initial
;                    ||||||||
;                    |||||||+-- IRQ Edge/Level (1=falling/low level, 0=falling)
;                    ||||||+--- IRQ interrupt mask (1=IRQ interrupt mask)
;                    |||||+---- IRQ interrupt acknowledge (1=clear IRQ int flag)
;                    ||||+----- IRQ interrupt flag (1=pending)
;                    ++++------ NIL=0 (bit 7-4)
;
```

```
HDB_Init          equ %00000010     ; HDB initial
;                     ||||||||
;                     |||||||+-- PPI clock[0] [1:0]= 10 = 32.768KHz Xtal
;                     ||||||+--- PPI clock[1] [1:0]= 01 = external PPIECLK pin
;                     |||||+---- HDB2 (1= PTB2 is high current)
;                     ||||+----- HDB3 (1= PTB3 is high current)
;                     ||++------ b4: NIL=0 (bit 4)
;                     |+-------- PPI1 interrupt flag (1=pending)
;                     +--------- NIL=0 (bit 7)
;                      *Remark: PPIECLK mux with PTB2 & T1CH0 !!!!
;
KBSCR_Init        equ %00000100     ; KBSCR initial
;                     ||||||||
;                     |||||||+-- KBI Edge/Level (1=falling/low level, 0=falling)
;                     ||||||+--- KBI/PPI interrupt mask (1=KBI/PPI int mask)
;                     |||||+---- KBI/PPI interrupt ack(1=clear KBI/PPI int flag)
;                     ||||+----- KBI interrupt flag (1=pending)
;                     ++++------ NIL=0 (bit 7-4)
; * Remark: For KBI int, needs to check KBI interrupt flag in KBSCR !!!
; * Remark: For PPI int, needs to check PPI1 interrupt flag in HDB
;
KBIER_Init        equ %00000111     ; KBIER initial (KBI Enable)
;                     ||||||||
;                     |||||||+-- KBIE0 (1= PTA0 KBI interrupt enable)
;                     ||||||+--- KBIE1 (1= PTA1 KBI interrupt enable)
;                     |||||+---- KBIE2 (1= PTA2 KBI interrupt enable)
;                     ||||+----- KBIE3 (1= PTA3 KBI interrupt enable)
;                     |||+------ PPI1E[0] (000)=PPI disable, (001)= 512 count
;                     ||+------- PPI1E[1] (010)=1024, (011)= 2048, (100)= 4096
;                     |+-------- PPI1E[2] (101)=8192, (110)= 16834, (111)= 32768
;                     +--------- NIL=0 (bit 7)
; * Remark: When KBIEx = 1, internal pullup enable and PTAx force to i/p
;
LCDCR_Init        equ %10010000     ; LCDCR initial (LCD Enable)
;                     ||||||||
;                     |||||||+-- LCCON0 (LCCON3:0 is bias vol control)
;                     ||||||+--- LCCON1
;                     |||||+---- LCCON2
;                     ||||+----- LCCON3
;                     |||+------ LC (LC =1 low current, FC =1 fast charge)
;                     ||+------- FC (FC:LC) x:0=37K, 0:1=146K, 1:1=Fast Charge
;                     |+-------- NIL=0 (bit 6)
;                     +--------- LCDE (1= LCD enable)
;
LCDCLK_Init       equ %00010001     ; LCDCLK initial
;                     ||||||||
;                     |||||||+-- LCLK0 000=256Hz, 001=128Hz, 010=64Hz, 011=32Hz
;                     ||||||+--- LCLK1 Frame rate = LCDCLK * Duty cycle
;                     |||||+---- LCLK2 with LCDCLK=256Hz => 256*(1/4)=64Hz
;                     ||||+----- DUTY[0] 00=Static, 01=1/3 duty cycle
;                     |||+------ DUTY[1] 10=1/4 duty cycle, 11=Not used
;                     ||+------- FCCTL[0] 00=LCDCLK/32, 01=/64, 10=/128 w/LC=0
;                     |+-------- FCCTL[1] (Fast charge duty cycle)
;                     +--------- NIL=0 (bit 7)
```

```
;
; ---------------------------------------------------------------------------- *
; Define by 4MHz XTAL (1MHz bus) = 1uS bus clock
; ---------------------------------------------------------------------------- *
; Timer 1 Related Constant: 125mS timer overflow interrupt (Replace PPI)

PwmPeriodH1      equ $3D       ; TOF period = 125mS = 125000uS/8 =15625uS=$3D09
PwmPeriodL1      equ $09       ; Timer 1 active after MCU wakeup from KBI & stop

; ---------------------------------------------------------------------------- *
; Timer 2 Related Constant: 8uS duty cycle, 26uS Carrier freq
; to provide 38.64Khz carrier with 30.8% duty cycle(ON Tx diode) Logic1=ON state
; 1MHz = 1uS => PWM period = 26uS = $19, Duty = 8uS = $07 (1uS for compensation)

PwmPeriodH2      equ $00       ; Define PWM period = 26uS = $1A-1 = $19
PwmPeriodL2      equ $19       ; (i.e. PWM freq = 38.64KHz carrier)
DutyCycleH2      equ $00       ; Define 1/3 duty cycle = 8uS = $08-1 = $07
DutyCycleL2      equ $07       ; (i.e. 30.8% duty ON cycle)
; ---------------------------------------------------------------------------- *
; PART II: System Related values and variables
; ---------------------------------------------------------------------------- *
Auto_Mode_Init   equ %10100000 ;25oC, Sleep_off, swing_off, auto_wind
Heat_Mode_Init   equ %11010000 ;28oC, Sleep_off, swing_off, auto_wind
Tx_Flag_Init     equ %00100011 ;TX_READY=0, TX_CNT=35 (Count down)

Data10_Init      equ %00000010 ;AC_OFF, Auto_mode, oC, Light ON, Model
Data32_Init      equ %10100000 ;25oC, Sleep_off, Swing_off, auto_wind
Data54_Init      equ %00001000 ;Model Set to Model 0 (b0 always equal to 0)
CtmCode_Init     equ %10101001 ;0.63ms low + 0100101 customer code (Tx LSB first)

;(Value for Tx frame delay call for 0.5mS delay) for FSL
Head_Time_ON     equ $10       ; Carrier on time for heading (8mS) 16*0.5ms
Head_Time_OFF    equ $08       ; Carrier off time for heading (4mS) 8*0.5ms

; 0us for compensation of time delay by the instruction delay
Data0_Time_ON    equ $32       ; Carrier on time for data 0 (500uS) 50*10us
Data0_Time_OFF   equ $32       ; Carrier off time for data 0 (500uS) 50*10us
Data1_Time_ON    equ $32       ; Carrier on time for data 1 (500uS) 50*10us
Data1_Time_OFF   equ $96       ; Carrier off time for data 1 (1500uS) 150*10us
; ---------------------------------------------------------------------------- *
; Key_Flag bit definition
KEY_ON           equ 7         ;=1 if KBI occur, =0 if key released
KEY_WRONG        equ 6         ;=1 if Key Wrong, =0 if Key O.K.
KEY_FIRST_ON     equ 5         ;=1 if first timer setting ON key pressed
KEY_CONFIRM      equ 4         ;=1 if Model # confirm key pressed
LCD_READY                              equ 3         ;=1 go to LCD routine
KEY_REP          equ 2         ;=1 if Key Repeat within 250mS
TIM_FLASH        equ 1         ;=1 if toggle in 250mS T1OF
S34_KEY_ON       equ 0         ;=1 if S3 or S4 pressed

; Tx_Flag bit definition

TX_READY         equ 7         ;=1 if Tx ready, =0 if Tx not ready
```

```
; ------------------------------------------------------------------------- *
T_Flash_Init     equ $30           ; Timer_Flash_Cnt initial 125mS*48=6s(48=$30)
S34_Cnt_Init     equ $18           ; S3 & S4 cnt initial 125mS*24=3s(24=$18)
Key_Sleep_Max    equ $50           ; Key_Sleep_Cnt initial 125mS*80=10s(80=$50)
; ------------------------------------------------------------------------- *

DEFAULT_RAM                    SECTION SHORT

;       org      RamStart  ; $0080 ($80-$8F)
Key_Flag         ds  1     ; KEY_ON flag (1=key interrupt occur)
Key_Value        ds  1     ; Store Key value(ON/OFF, +, -, ...etc)
                           ; S1=%11101110,S2=%11101101
Key_Sleep_Cnt    ds  1     ; Store Key SLEEP_CNT value
Timer_Flash_Cnt  ds  1     ; Timer setting flash counter(6sec decremental Timer)
                           ; 125mS * 48 = 6sec (48= $30)
Sleep_Flash_Cnt  ds  1     ; Sleep key flash counter (1s if sleep key press)

Auto_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%10010000
Cool_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%10010000
Humd_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%10010000
Wind_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%10010000
Heat_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%11000000

Tx_Data_Temp     ds  1     ; Temp store Tx_Data value
T_Pointer        ds  1     ; Index for MODEL# lookup table ($00 - $09)

Tx_Flag          ds  1     ; TX_READY Flag (1=ready) &
                           ; TX_CNT (35-bit of data) =%00100011=35

S34_Key_Cnt      ds  1     ; S3 & S4 key pressed counter
Tx_Data54_Tmp    ds  1     ; Temp store Tx_Data54
T_Pointer_Tmp    ds  1     ; Temp store T_Pointer

; ------------------------------------------------------------------ *
;       org       $0090     ; $90-$9F

Tx_Data10        ds  1     ; Nibble 1-0 (first 4-bit will be shift out)=%0000xxxx
Tx_Data32        ds  1     ; Nibble 3-2 =%10010000
Tx_Data54        ds  1     ; Nibble 5-4 =%00000001
Tx_CtmCode       ds  1     ; Nibble 9-8 (last bit will be shift out)=%x0100101

PPI_Cnt          ds  1     ; PPI counter for LCD flash when sleep key pressed

;*****************************************************************************
```

```
DEFAULT_ROM             SECTION

; ------------------------------------------------------------------------- *
; Program Area                                                              *
; ------------------------------------------------------------------------- *
;       org       RomStart  ; $DE00
; ------------------------------------------------------------------------- *
; Subroutine Initialization : Configure register, Port, KBI & Timer         *
; In      :  <nil>                                                          *
; Out     :  <nil>                                                          *
; Call    :  <nil>                                                          *
; ------------------------------------------------------------------------- *
;***************************************************************************
; Entry Point
;***************************************************************************
Entry:
main:
       rsp                ; initialize the stack pointer
       SEI                ; mask all interrupt
       sta $FFFF          ; Clear COP counter
; ------------------------------------------------------------------------- *
; Port Initial
; ------------------------------------------------------------------------- *
       mov #%00000111,PTA  ; Avoid false KBI by write PTA0-2 as o/p high
       mov #%11111111,DDRA ; Set all port to output low except PTA0-2
                           ; Set PTA4-6 o/p low, KBI will occur if any key
                           ; pressed after KBI interrupt enable

       clr PTB             ; Set all port to output low
       clr PTC
       clr PTD
       clr PTE

       lda #$FF            ; Change port data register before change data
       sta DDRB            ; direction register (aviod glitch)
       sta DDRC
       sta DDRD
       sta DDRE
       sta PTD             ; enable LCD output
       sta PTE             ; enable LCD output
;***************************************************************************
; ------------------------------------------------------------------------- *
; Configuration Register Initial
; ------------------------------------------------------------------------- *
       mov #CONFIG1_Init,CONFIG1   ; CONFIG1 Initial
       mov #CONFIG2_Init,CONFIG2   ; CONFIG2 Initial
       lda #LVISR_Init
       sta LVISR                   ; LVI Initial
       mov #INTSCR_Init,INTSCR     ; IRQ Initial
```

```
; ------------------------------------------------------------------------ *
; Keyboard Interrupt & PPI Initial
; ------------------------------------------------------------------------ *
       mov #HDB_Init,HDB           ; PTB & PPI Clock Initial
       mov #KBSCR_Init,KBSCR       ; KBI & PPI Initial
       mov #KBIER_Init,KBIER       ; KBI & PPI Interrupt Initial
; ------------------------------------------------------------------------ *
; Timer 1 & 2 Initial
; ------------------------------------------------------------------------ *
; Set T1CH0 Overflow (125ms) to replace PPI periodic interrupt
;                              ; Set T1CH0 & T1CH1 as port
;                              ; T1SC0 & T1SC1 = Default value $00
;

       mov #%01110011,T1SC     ; TOF int Enable, Timer stop & reset, clock=bus/8
       mov #PwmPeriodH1,T1MODH ; write overflow value to modulo reg (PWM freq)
       mov #PwmPeriodL1,T1MODL
; ---------------------------------------------------------------- *
; Set T2CH0 as Output compare + Overflow (38K carrier) w/o interrupt
                              ; Set T2CH1 as port, T2SC1=default value =$00

       mov #PwmPeriodH2,T2MODH ; write overflow value to modulo reg (PWM freq)
       mov #PwmPeriodL2,T2MODL

       mov #DutyCycleH2,T2CH0H ; write output compare value (Duty cycle)
       mov #DutyCycleL2,T2CH0L
; ------------------------------------------------------------------------ *
; Keyboard status and Mode & Tx_Data Initial
; ------------------------------------------------------------------------ *
       clr Key_Flag            ; Initial Keyboard flag & value and counter
       clr Key_Value           ; S1=%11101110,S2=%11101101...etc
       clr Key_Sleep_Cnt       ; Initial Sleep counter
       clr T_Pointer           ; Initial Timer Setting Pointer
       clr Sleep_Flash_Cnt     ; Initial Timer flash counter
       clr Tx_Flag             ; Initial Tx flag
       clr PPI_Cnt

       lda #Auto_Mode_Init     ; Initial difference mode value
                               ; wind speed + Temperature (Tx_Data32)
       sta Auto_Mode
       sta Cool_Mode
       sta Humd_Mode
       sta Wind_Mode
       mov #Heat_Mode_Init,Heat_Mode

       clr Tx_Flag

       mov #Data10_Init,Tx_Data10       ; Initial Tx Data + Customer code
       mov #Data32_Init,Tx_Data32
       mov #Data54_Init,Tx_Data54
       mov #CtmCode_Init,Tx_CtmCode

       mov Tx_Data54,Tx_Data54_Tmp     ; Sync Tx_Data54 & Tx_Data54_Tmp
       mov T_Pointer,T_Pointer_Tmp     ; Sync T_Pointer & T_Pointer_Tmp
```

```
; ------------------------------------------------------------------------ *
; LCD Data & Configuration Register Initial
; ------------------------------------------------------------------------ *
       clrh
       clrx
       lda #$FF                ; default LCD pattern (All ON)
Ld_LCD_FF:
       sta LDAT1,x
       incx
       cpx #$0D
       blo Ld_LCD_FF

       mov #%10101111,LDAT9    ; OFF unused LCD segments
       mov #%10000000,LDAT5
       bclr 4,LDAT4

       mov #LCDCR_Init,LCDCR   ; Initial LCD
       mov #LCDCLK_Init,LCDCLK

       jsr Delay_500ms         ; All segment in LCD panel ON (3 sec)
       jsr Delay_500ms
       jsr Delay_500ms
       jsr Delay_500ms
       jsr Delay_500ms
       jsr Delay_500ms

; ------------------------------------------------------------------- *
       clrh
       clrx
       lda #$00        ; default LCD pattern(All OFF except 25oC + Light)
Ld_LCD_00:
       sta LDAT1,x
       incx
       cpx #$0A                ; Reach to LDAT11?
       bls Ld_LCD_00

       mov #%11101011,LDAT4    ; "MODEL 0" segment ON
       bset 7,LDAT5

       lda #%11010111          ; "25oC", "TEMP" & "Light" segment ON
       sta LDAT8
       lda #%10101101
       sta LDAT7
       bset 0,LDAT3

; ------------------------------------------------------------------------ *
; Remote Run after power on reset (POR) --- Waiting for Keyboard Interrupt
; ------------------------------------------------------------------------ *
Main_Loop:

       mov #T_Flash_Init,Timer_Flash_Cnt   ; Initial Timer flash counter
       mov #S34_Cnt_Init,S34_Key_Cnt       ; Initial S3 S4 timer counter
```

```
        CLI                     ; Enable all interrupts (i.e. waiting for KBI)
        stop                    ; And PPI disable

        jsr TIM1_RUN            ; Enable Timer 1 TOF interrupt and timer ON

        bra First_Key

Repeat_Key_Scan:

        wait                    ; Waiting for 125ms TOF / KBI

First_Key:

        bset LCD_READY,Key_Flag
        jsr K_Scan              ; Jump to keyboard scan (KEY_ON, KEY_WRONG)
                                ; KEY_ON, KEY_WRONG, Key_Value, & Tx_Data updated

        bclr 1,LDAT9            ; Tx signal OFF (A2,A3,A4)
        bclr 2,LDAT9
        bclr 5,LDAT9

Slip_Key_Scan:
                                ; Update LCD in each 125ms (T1OF)
        jsr LCD_Update          ; Update LCD(Always update even no key, wrong key)
                                ; Because it provide flash in sleep key & timer
                                ; setting

        jsr Key_Release         ; Check Key released (KEY_ON, TX_READY)
                                ; PTA=%00001111 All Key standby KBI

        jsr Tx_Frame            ; Frame Transmission if KEY_ON=0 & TX_READY=1

; ------------------------------------------------------- *
        lda Key_Sleep_Cnt
        cmp #Key_Sleep_Max       ; 80 x 125ms = 10sec
        blo Repeat_Key_Scan              ; If no key press within 10sec

        clr Key_Sleep_Cnt        ; Reset Sleep Counter

        mov #%00000111,PTA       ; Avoid false KBI by write PTA0-2 as o/p high
                                 ; Set PTA4-6 o/p low for KBI wakeup

        brset 3,Tx_Data32,Sleep_PPI

        stop                     ; Waiting for any key to wakeup from stop mode

        bra First_Key

Sleep_PPI:

        clr PPI_Cnt

        mov #%01110111,KBIER     ; Enable PPI before enter STOP mode
```

```
Stop_N2_Flash:
       bclr 0,LDAT9                ; A1 OFF
       bset 3,LDAT9                ; A5 ON
       stop
       brset 7,PPI_Cnt,Stop_N2_End  ; bit-7 of PPI_Cnt will set if KBI occur
                                    ; (i.e. Key pressed)

       bset 0,LDAT9                ; A1 ON
       bclr 3,LDAT9                ; A5 OFF
                         stop
       brset 7,PPI_Cnt,Stop_N2_End

       bset 0,LDAT9                ; A1 ON
       bset 3,LDAT9                ; A5 ON
                         stop
       brset 7,PPI_Cnt,Stop_N2_End

       bra Stop_N2_Flash

Stop_N2_End:

       mov #%00000111,KBIER     ; Disable PPI before exit STOP mode

       clr PPI_Cnt

       bra First_Key

; ------------------------------------------------------------------------------ *


; ------------------------------------------------------------------------------ *
; Subroutine <500ms Delay>                                                       *
; Bus Clock = 1MHz, 1 Cycle=1uS                                                  *
; In      :  <nil>                                                               *
; Out     :  <nil>                                                               *
; Call    :  <nil>                                                               *
; ------------------------------------------------------------------------------ *
Delay_500ms:
       LDX #$C8                ; [2]200
Delay_Yms_X
       LDA #$FA                ; [2]250
Delay_Yms_A
       sta $FFFF               ; [4] clear COP
       nop                     ; [1]
       nop                     ; [1]
       nop                     ; [1]
       dbnza Delay_Yms_A       ; [3] 10*{A}=10A
       dbnzx Delay_Yms_X       ; [3] 10A*{X}+2+3=10AX+2+3
       rts                     ; [3]

;Total= {[2]+[2]+10AX+[3]}*Bus Cycle
;Total= [7+(10*200*250)]*1uS = 500mS
```

```
; ------------------------------------------------------------------------ *
; Timer 1 TOF enable --- 200ms
; ------------------------------------------------------------------------ *
TIM1_RUN:                       ; Enable TIM1
        bclr 5,T1SC             ; clear TSTOP, enable timer counter
        bclr CH0F,T1SC0         ; clear TIMER1 CH0F flag
        bclr CH1F,T1SC1         ; clear TIMER1 CH1F flag
        bclr TOF,T1SC           ; clear TIMER1 TOF flag
        rts

; ------------------------------------------------------------------------ *
; Timer 1 TOF disable --- 200ms
; ------------------------------------------------------------------------ *
TIM1_STOP:                      ; Disable TIM1
       mov #%01110110,T1SC      ; TOF int Enable, Timer stop & reset, clock=bus/64
       rts

; ------------------------------------------------------------------------ *
; Key Release Check
; ------------------------------------------------------------------------ *
Key_Release:
       mov #%00001111,PTA       ; PTA4-7=0 (Check All Key in PTA3-0)
       nop
       lda PTA
       and #%00001111           ; mask upper nibble
       cmp #%00001111
       beq K_Released           ; Key Released
       bset KEY_ON,Key_Flag     ; Set key on flag if any key pressed
       bset KEY_REP,Key_Flag    ; Set repeat key flag
       rts

K_Released:
       bclr KEY_ON,Key_Flag     ; Clear Key pressed flag if no key pressed
       bclr KEY_REP,Key_Flag          ; Clear repeat key flag
       bclr S34_KEY_ON,Key_Flag
       mov #S34_Cnt_Init,S34_Key_Cnt  ; Initial S3 S4 timer counter
       rts

; ------------------------------------------------------------------------ *
; Subroutine <Hms Delay>                                                   *
; Bus Clock = 1MHz, 1 Cycle=1uS                                            *
; In      :  Acc                                                           *
; Out     :  <nil>                                                         *
; Call    :  <nil>                                                         *
; Remark  : (0.5mS * A) Delay                                              *
; ------------------------------------------------------------------------ *
Delay_Hms:

Delay_Hms_A:
       LDX #$31                 ; [2]49
Delay_Hms_X:
       sta $FFFF                ; [4] clear COP !!!!!!! Need to re-!!!
       nop                      ; [1]
       nop                      ; [1]
```

```
        nop                     ; [1]
        dbnzx Delay_Hms_X       ; [3] 10*{X}=10X
        dbnza Delay_Hms_A       ; [3] 10X*{A}+2+3=10AX+5
        rts                     ; [3] 2+10XA+5+3=10XA+10

;Total= {10AX+[10]}*Bus Cycle
;Total= [10+(10*49*A)]*1uS = 500uS*A =A*0.5mS

; ------------------------------------------------------------------------- *
; Keyboard Scan Routine                                                     *
; In      :  <Key_Flag>                                                     *
; Out     :  <Key_Value>                                                    *
; Call    :  <nil>                                                          *
; ------------------------------------------------------------------------- *
K_Scan:                         ; Key location and expect KEY_ON=1

        lda #$0A                ; 10*0.5ms
        jsr Delay_Hms           ; Delay 5ms for Key Debounce

        bclr KEY_WRONG,Key_Flag         ; Clear wrong key flag
        brset KEY_ON,Key_Flag,KS_S1_S3 ; Key Scan if key pressed
        jmp Wrong_Key                   ; No Key detected in KBI & KEY_ON=0,
                                        ; (Key released) may be due form niose

KS_S1_S3:
        mov #%01100111,PTA      ; PTA4=0 (Check S1-S3), PTA3 & 7 o/p low
        mov #%11111000,DDRA
        lda PTA
        sta Key_Value           ; Store Key Value
        and #%00000111          ; mask bit3-7
        cmp #%00000111
        beq KS_S4_S6            ; No key if equal

        lda #$0A                ; 10*0.5ms
        jsr Delay_Hms           ; Delay 5ms

        lda PTA
        cmp Key_Value
        bne Wrong_Key           ; Wrong Key if not equal
        clr Key_Sleep_Cnt       ; Reset Key Sleep Counter if any key detected
        bra K_S1_S3             ; Go to S1-S3 Key decode

KS_S4_S6:
        mov #%01010111,PTA      ; PTA5=0 (Check S4-S6), PTA3 & 7 o/p low
        lda PTA
        sta Key_Value           ; Store Key Value
        and #%00000111          ; mask bit3-7
        cmp #%00000111
        beq KS_S7_S9            ; No key if equal

        lda #$0A                ; 10*0.5ms
        jsr Delay_Hms           ; Delay 5ms

        lda PTA
```

```
        cmp Key_Value
        bne Wrong_Key           ; Wrong Key if not equal
        clr Key_Sleep_Cnt       ; Reset Key Sleep Counter if any key detected
        bra K_S4_S6             ; Go to S4-S6 Key decode

KS_S7_S9:
        mov #%00110111,PTA      ; PTA6=0 (Check S7-S9), PTA3 & 7 o/p low
        lda PTA
        sta Key_Value           ; Store Key Value
        and #%00000111          ; mask bit3-7
        cmp #%00000111
        beq Wrong_Key           ; No key in S1-S9 if equal (all no key=wrong key)

        lda #$0A                ; 10*0.5ms
        jsr Delay_Hms           ; Delay 5ms

        lda PTA
        cmp Key_Value
        bne Wrong_Key           ; Wrong Key if not equal
        clr Key_Sleep_Cnt       ; Reset Key Sleep Counter if any key detected
        bra K_S7_S9             ; Go to S7-S9 Key decode

Wrong_Key:

        bset KEY_WRONG,Key_Flag ; Assume No key press OR noise
        bclr TX_READY,Key_Flag
        rts
; ------------------------------------------------------------------------- *
; Key Decode
; ------------------------------------------------------------------------- *
K_S1_S3:
        brclr 0,Key_Value,S1x   ; Perfrom S1 (OK) "Sleep" action
        brclr 1,Key_Value,S2x   ; Perfrom S2 (A.M.WIND) action
        brclr 2,Key_Value,S3x   ; Perfrom S3 (SET) "Model select" action

K_S4_S6:
        brclr 0,Key_Value,S4x   ; Perfrom S4 (+) action [oC / Model set]
        brclr 1,Key_Value,S5x   ; Perfrom S5 (M.WIND) "Light ON/OFF" action
        brclr 2,Key_Value,S6x   ; Perfrom S6 (ON/OFF) action

K_S7_S9:
        brclr 0,Key_Value,S7x   ; Perfrom S7 (-) action [oC / Model set]
        brclr 1,Key_Value,S8x   ; Perfrom S8 (WIND) "Fan Speed select" action
                                ; [->Auto->Min->Mid->Max->]Fan Speed
        brclr 2,Key_Value,S9x   ; Perfrom S9 (MODE) action
                                ; [->Auto->cool->dry->wind->heat->]modes


        bra Wrong_Key           ; If any mistake in key scan

                                ; Jump to Tx Data Update
S1x:    brset KEY_REP,Key_Flag,Rep_Key
        jmp S1
S2x:      brset KEY_REP,Key_Flag,Rep_Key
```

```
        jmp S2
S3x:    brset KEY_REP,Key_Flag,Rep_Key
        jmp S3
S4x:    brset KEY_REP,Key_Flag,S4_Rep_Key  ; S4 No Repeat Key
        jmp S4
S5x:      brset KEY_REP,Key_Flag,Rep_Key
        jmp S5
S6x:      brset KEY_REP,Key_Flag,Rep_Key
        jmp S6
S7x:      brset KEY_REP,Key_Flag,S7_Rep_Key  ; S7 No Repeat Key
        jmp S7
S8x:      brset KEY_REP,Key_Flag,Rep_Key
        jmp S8
S9x:      brset KEY_REP,Key_Flag,Rep_Key
        jmp S9

Rep_Key:
        rts

; ------------------------------------------------------------------------- *
S4_Rep_Key:

        lda S34_Key_Cnt
        beq S4y               ; Go to S4 decode if S4/S7 pressed > 3sec
        bra Rep_Key

S7_Rep_Key:

        lda S34_Key_Cnt
        beq S7y               ; Go to S7 decode if S4/S7 pressed > 3sec
        bra Rep_Key

S4y:    jmp S4
S7y:    jmp S7

; ------------------------------------------------------------------------- *
; Tx Data Update
; ------------------------------------------------------------------------- *
S1:                                ; Sleep Key pressed (OK)
        brclr 7,Tx_Data10,Slip_S1  ; No action if S1=OFF

        brset 3,Tx_Data32,Sleep_OFF ; Check ON/OFF? (1=ON)

                                   ; Here AC ON
        lda Tx_Data10
        and #%01110000             ; mask all bit except b6-4
        cmp #%00010000             ; Check Cool mode (001)
        beq Sleep_ON_Cool
        cmp #%00100000             ; Check Humd mode (010)
        beq Sleep_ON_Humd
        cmp #%01000000             ; Check Heat mode (100)
        beq Sleep_ON_Heat
        bra Slip_S1                ; Slip if in others modes
```

```
Sleep_ON_Cool:
       bset 3,Cool_Mode
       bra Set_Data32
Sleep_ON_Humd
       bset 3,Humd_Mode
       bra Set_Data32

Sleep_ON_Heat
       bset 3,Heat_Mode

Set_Data32:
       bset 3,Tx_Data32            ; OFF -> ON
       bset TX_READY,Tx_Flag       ; Tx ready
       bra End_S1

Sleep_OFF:                         ; Here AC OFF

       lda Tx_Data10
       and #%01110000              ; mask all bit except b6-4
       cmp #%00010000              ; Check Cool mode (001)
       beq Sleep_OFF_Cool
       cmp #%00100000              ; Check Humd mode (010)
       beq Sleep_OFF_Humd
       cmp #%01000000              ; Check Heat mode (100)
       beq Sleep_OFF_Heat
       bra Slip_S1

Sleep_OFF_Cool:
       bclr 3,Cool_Mode
       bra Clr_Data32
Sleep_OFF_Humd
       bclr 3,Humd_Mode
       bra Clr_Data32

Sleep_OFF_Heat
       bclr 3,Heat_Mode

Clr_Data32:
       bclr 3,Tx_Data32            ; OFF -> ON
       bset TX_READY,Tx_Flag       ; Tx ready
       bra End_S1

Slip_S1:
End_S1:
       rts
; ------------------------------------------------------------------------- *
S2:                                ; Swing Key pressed (A.M.WIND)
       brclr 7,Tx_Data10,Slip_S2   ; No action if S1=OFF

                                   ; Here (S1=ON)
       brset 2,Tx_Data32,Swing_OFF ; Check ON/OFF? (1=ON)
       bset 2,Tx_Data32            ; OFF -> ON
       bset 2,Auto_Mode
```

```
        bset 2,Cool_Mode
        bset 2,Humd_Mode
        bset 2,Wind_Mode
        bset 2,Heat_Mode
        bset TX_READY,Tx_Flag       ; Tx ready
        bra End_S2

Swing_OFF:
        bclr 2,Tx_Data32            ; ON -> OFF
        bclr 2,Auto_Mode
        bclr 2,Cool_Mode
        bclr 2,Humd_Mode
        bclr 2,Wind_Mode
        bclr 2,Heat_Mode
        bset TX_READY,Tx_Flag       ; Tx ready

Slip_S2:
End_S2:
        rts
; ------------------------------------------------------------------------- *
S3:                                 ; "Model Set" Key pressed (SET)

        brclr 3,Tx_Data54,Model_Set

Model_Confirm:
        bclr 3,Tx_Data54            ; Model Confrim (MODEL ON)
        bset KEY_CONFIRM,Key_Flag
        rts

Model_Set:
        bset 3,Tx_Data54            ; Model Set (MODEL flash)
        bclr KEY_CONFIRM,Key_Flag
        mov #T_Flash_Init,Timer_Flash_Cnt
        mov Tx_Data54,Tx_Data54_Tmp
        mov T_Pointer,T_Pointer_Tmp
        rts
; ------------------------------------------------------------------------- *
S4:                                 ; + Key pressed for oC / Model Set (^)
        brset 3,Tx_Data54,Model_Set_S4 ; Check Timer set status (4.3=1 ?)
S4_Normal:
        brclr 7,Tx_Data10,Slip_S4   ; No action if S1=OFF
        lda Tx_Data10
        and #%01110000
        beq Slip_S4             ; No action if in Auto Mode

        lda Tx_Data32
        nsa                     ; swrap N3 as lower nibble
        sta Tx_Data_Temp
        and #%00001111          ; mask N2
        cmp #%00001111          ; Is reach Max $111, No need Inc
        bhs No_Inc_Data32

        inc Tx_Data_Temp
        lda Tx_Data_Temp
```

```
        nsa                     ; Swrap N3 back to higher nibble
        sta Tx_Data32

No_Inc_Data32:
        jsr Data32_To_Modes     ; Check & update Data32 & Modes
        bset TX_READY,Tx_Flag   ; Tx ready
        bset S34_KEY_ON,Key_Flag

Slip_S4:
End_S4:
        rts

Model_Set_S4:                    ; Here Timer setting = ON
        brset KEY_CONFIRM,Key_Flag,S4_Normal ; No action if Key Confirm
        lda Timer_Flash_Cnt
        beq End_S4
        mov #T_Flash_Init,Timer_Flash_Cnt   ; Reset Timer flash counter
        clrh
        ldx T_Pointer           ; load Timer pointer
        cpx #$09                ; Reach Max Model 9?
        blo Inc_T_P             ; Lower than Model 9
        clrx                    ; Reset Timer Pointer to Min(Model 0)
        bra Up_Data54
Inc_T_P:
        incx                    ; Increase Timer setting
Up_Data54:
        stx T_Pointer           ; Update T_Pointer
        lda $F030,x             ; Load Data54 Table
        sta Tx_Data54           ; Update Data54 (4.3=1)
        bra End_S4
; ------------------------------------------------------------------------------ *
S5:                             ; Light Key pressed (M.WIND)
        brset 1,Tx_Data10,Light_OFF ; Check ON/OFF? (1=ON)
        bset 1,Tx_Data10            ; OFF -> ON
        bset TX_READY,Tx_Flag       ; Tx ready
        bra End_S5

Light_OFF:
        bclr 1,Tx_Data10            ; ON -> OFF
        bset TX_READY,Tx_Flag       ; Tx ready

End_S5:
        rts
; ------------------------------------------------------------------------------ *
S6:                             ; ON/OFF Key pressed   (ON/OFF)
        brclr 7,Tx_Data10,S6_ON ; Check ON/OFF status

        bclr 7,Tx_Data10        ; Change to OFF state
        bra Clear_T_S

S6_ON:
        bset 7,Tx_Data10        ; Change to ON state
```

```
Clear_T_S:
       bclr 3,Tx_Data32        ; Clear Sleep
       bclr 3,Auto_Mode
       bclr 3,Cool_Mode
       bclr 3,Humd_Mode
       bclr 3,Wind_Mode
       bclr 3,Heat_Mode
       bset TX_READY,Tx_Flag   ; Tx ready

       rts
; ------------------------------------------------------------------------------ *
S7:                                ; - Key pressed for oC / Model Set (v)
       brset 3,Tx_Data54,Model_Set_S7 ; Check Timer set status (4.3=1 ?)
S7_Normal:
       brclr 7,Tx_Data10,Slip_S7     ; No action if S1=OFF
       lda Tx_Data10
       and #%01110000
       beq Slip_S7             ; No action if in Auto Mode

       lda Tx_Data32
       nsa
       sta Tx_Data_Temp        ; swrap N3 as lower nibble
       and #%00001111          ; mask N2
       cmp #%00000000          ; Is reach Min $0000, No need Dec
       beq No_Dec_Data32

       dec Tx_Data_Temp
       lda Tx_Data_Temp
       nsa                     ; Swrap N3 back to higher nibble
       sta Tx_Data32

No_Dec_Data32:
       jsr Data32_To_Modes     ; Check & update Data32 & Modes
       bset TX_READY,Tx_Flag   ; Tx ready
       bset S34_KEY_ON,Key_Flag

Slip_S7:
End_S7:

       rts

Model_Set_S7:                   ; Here Timer setting = ON
       brset KEY_CONFIRM,Key_Flag,S7_Normal ; No action if Key Confirm
       lda Timer_Flash_Cnt
       beq End_S7
       mov #T_Flash_Init,Timer_Flash_Cnt   ; Reset Timer flash counter
       clrh
       ldx T_Pointer           ; load Timer pointer
       cpx #$00                ; Reach Max Model 0?
       bhi Dec_T_P             ; Lower than Model 24
       ldx #$09                ; Reset Timer Pointer to Max(Model 9)
       bra Down_Data54
Dec_T_P:
       decx                    ; Decrease Timer setting
```

```
Down_Data54:
       stx T_Pointer           ; Update T_Pointer
       lda $F030,x             ; Load Data54 Table
       sta Tx_Data54           ; Update Data54 (4.3=1)
       bra End_S7
; ------------------------------------------------------------------------- *
S8:                            ; "Fan Speed" Key pressed (WIND)
                               ;(Auto>low>mid>high)
       brclr 7,Tx_Data10,Slip_S8 ; No action if AC OFF

       lda Tx_Data32           ; Here (AC ON)
       and #%00000011          ; mask other bit except b1-0
       cmp #%00000011          ; Reach Max. value (high)?
       beq Rst_Wind
Inc_Wind:
       inc Tx_Data32
       bra End_S8

Rst_Wind:
       bclr 0,Tx_Data32
       bclr 1,Tx_Data32        ; Change to Min. value (Auto)

End_S8:
       jsr Data32_To_Modes     ; Check & update Data32 & Modes
       bset TX_READY,Tx_Flag   ; Tx ready

Slip_S8:
       rts
; ------------------------------------------------------------------------- *
S9:                            ; Modes Key pressed (MODE)
                               ; (Auto>Cool>Humd>Wind>Heat)
       brclr 7,Tx_Data10,Slip_S9 ; No action if S1=OFF

       bclr 3,Tx_Data32        ; Clear Sleep
       bclr 3,Auto_Mode
       bclr 3,Cool_Mode
       bclr 3,Humd_Mode
       bclr 3,Wind_Mode
       bclr 3,Heat_Mode
       lda Tx_Data10           ; Here (S1=ON)
       nsa
       and #%00001111          ; mask other bit except b4-0 (b4==1)
       cmp #%00001100          ; Reach Max. value (Heat mode)?
       blo Inc_Modes
       mov #%10000000,Tx_Data10 ; Change to Min. value (Auto mode)
       bset TX_READY,Tx_Flag   ; Tx ready
       bra End_S9

Inc_Modes:
       inca
       nsa
       sta Tx_Data10
       bset TX_READY,Tx_Flag   ; Tx ready
```

```
Slip_S9:
End_S9:
       rts
; ------------------------------------------------------------------------ *
; Update Data32 to Difference Modes (Auto mode check can be remove)
; ------------------------------------------------------------------------ *
Data32_To_Modes:

       lda Tx_Data10
       and #%01110000
       cmp #%00000000          ; Check Auto mode?
       beq D32_2_Auto
       cmp #%00010000          ; Check Cool mode?
       beq D32_2_Cool
       cmp #%00100000          ; Check Humd mode?
       beq D32_2_Humd
       cmp #%00110000          ; Check Wind mode?
       beq D32_2_Wind
       mov Tx_Data32,Heat_Mode ; It is Heat mode
       rts
D32_2_Auto:
       mov Tx_Data32,Auto_Mode
       rts
D32_2_Cool:
       mov Tx_Data32,Cool_Mode
       rts
D32_2_Humd:
       mov Tx_Data32,Humd_Mode
       rts
D32_2_Wind:
       mov Tx_Data32,Wind_Mode
       rts
; ------------------------------------------------------------------------ *
; LCD Data Update Subroutine (Depend on key value and then Tx Data)
; ------------------------------------------------------------------------ *
LCD_Update:

       brclr LCD_READY,Key_Flag,Slip_LCD  ; Update LCD in each 250ms (T1OF)

       bclr LCD_READY,Key_Flag ; Clear LCD_READY Flag

       jsr LCD_N1_Chk          ; LCD based on N1
       jsr LCD_N2_Chk          ; LCD based on N2
       jsr LCD_N3_Chk          ; LCD based on N3
       jsr LCD_N45_Chk         ; LCD based on N4 & N5
       jsr LCD_N67_Chk         ; LCD based on N6 & N7

Slip_LCD:
       rts
; ------------------------------------------------------------------------ *
LCD_N1_Chk:
       brclr 7,Tx_Data10,L_AC_OFF
                               ; Here AC ON
       lda Tx_Data10
```

```
        and #%01110000          ; Mask all bit except bit4-6
        cmp #%00000000          ; Is bit4-6 = %000
        beq LCD_Auto_Mode       ; Go to Auto mode if true
        cmp #%00010000          ; Is bit4-6 = %001
        beq LCD_Cool_Mode       ; Go to Cool mode if true
        cmp #%00100000          ; Is bit4-6 = %010
        beq LCD_Humd_Mode       ; Go to Humd mode if true
        cmp #%00110000          ; Is bit4-6 = %011
        beq LCD_Wind_Mode       ; Go to Wind mode if true
LCD_Heat_Mode:                  ; Otherwise go to Heat mode (%100 - %111)
        mov Heat_Mode,Tx_Data32 ; Here in Heat Mode
        bclr 3,LDAT10           ; AUTO OFF
        bclr 1,LDAT10           ; COOL OFF
        bclr 0,LDAT10           ; DRY OFF
        bclr 7,LDAT9            ; WIND OFF
        bset 2,LDAT10           ; HEAT ON
        rts
LCD_Auto_Mode:
        mov Auto_Mode,Tx_Data32 ; Here in Auto Mode
        bset 3,LDAT10           ; AUTO ON
        bclr 1,LDAT10           ; COOL OFF
        bclr 0,LDAT10           ; DRY OFF
        bclr 7,LDAT9            ; WIND OFF
        bclr 2,LDAT10           ; HEAT OFF
        rts
LCD_Cool_Mode:
        mov Cool_Mode,Tx_Data32 ; Here in Cool Mode
        bclr 3,LDAT10           ; AUTO OFF
        bset 1,LDAT10           ; COOL OFF
        bclr 0,LDAT10           ; DRY OFF
        bclr 7,LDAT9            ; WIND OFF
        bclr 2,LDAT10           ; HEAT ON
        rts
LCD_Humd_Mode:
        mov Humd_Mode,Tx_Data32 ; Here in Humd Mode
        bclr 3,LDAT10           ; AUTO OFF
        bclr 1,LDAT10           ; COOL OFF
        bset 0,LDAT10           ; DRY OFF
        bclr 7,LDAT9            ; WIND OFF
        bclr 2,LDAT10           ; HEAT ON
        rts
LCD_Wind_Mode:
        mov Wind_Mode,Tx_Data32 ; Here in Wind Mode
        bclr 3,LDAT10           ; AUTO OFF
        bclr 1,LDAT10           ; COOL OFF
        bclr 0,LDAT10           ; DRY OFF
        bset 7,LDAT9            ; WIND OFF
        bclr 2,LDAT10           ; HEAT ON
        rts
```

```
L_AC_OFF:
       clr LDAT3               ; Clear all except XXoC & Model No.
       clr LDAT6
       clr LDAT9
       clr LDAT10
       rts
; ------------------------------------------------------------------------- *
LCD_N2_Chk:
; -------------------------- *
LCD_N2_Sleep:

       brclr 3,Tx_Data32,L_Sleep_OFF

       bset 7,LDAT5            ; S6 always ON

       lda Sleep_Flash_Cnt
       cmp  #$03
       bls  A1_ON_A5_OFF     ; S8 ON & S7 OFF if ($00-$03)    [125ms*4=0.5s]
       cmp  #$07
       bls  A1_ON_A5_ON      ; S8 ON & S7 ON if ($04-$07)     [0.5s]
       cmp  #$0B
       bls  A1_OFF_A5_ON     ; S8 OFF & S7 ON if ($08-$0B)    [0.5s]
       bra LCD_N2_Swing

A1_OFF_A5_ON:
       bclr 0,LDAT9                ; A1 OFF
       bset 3,LDAT9                ; A5 ON
       bra LCD_N2_Swing

A1_ON_A5_OFF:
       bset 0,LDAT9                ; A1 ON
       bclr 3,LDAT9                ; A5 OFF
       bra LCD_N2_Swing

A1_ON_A5_ON:
       bset 0,LDAT9                ; A1 ON
       bset 3,LDAT9                ; A5 ON
       bra LCD_N2_Swing

L_Sleep_OFF:
       bclr 0,LDAT9                ; A1 OFF
       bclr 3,LDAT9                ; A5 OFF
; -------------------------- *
LCD_N2_Swing:
       brclr 2,Tx_Data32,L_Swing_OFF
       brclr 7,Tx_Data10,L_Swing_OFF ; Swing OFF if AC OFF
                                     ; Here AC ON

       bset 3,LDAT3             ; A.M.WIND ON
       bra LCD_N2_Wind_Speed
```

```
L_Swing_OFF:
       bclr 3,LDAT3             ; A.M.WIND OFF
; -------------------------- *
LCD_N2_Wind_Speed:

       brclr 7,Tx_Data10,L_Wind_OFF ; Wind Speed OFF if AC OFF
                                    ; Here AC ON
       lda Tx_Data32
       and #%00000011           ; Mask all except bit0-1
       cmp #%00000000           ; Is Bit0-1 = %00
       beq L_Wind_Auto          ; Go to Wind Auto if true
       cmp #%00000001           ; Is Bit0-1 = %01
       beq L_Wind_Low           ; Go to Wind Low if true
       cmp #%00000010           ; Is Bit0-1 = %10
       beq L_Wind_Mid           ; Go to Wind Mid if true
L_Wind_High:                    ; Otherwise go to Wind High (%11)
       bset 4,LDAT3             ; Max Wind ON
       mov #%00000111,LDAT6     ; AUTO OFF, FAN, Min Wind & Mid Wind ON
       rts
L_Wind_Low:
       bclr 4,LDAT3             ; Max Wind OFF
       mov #%00000110,LDAT6     ; FAN & Min Wind ON, AUTO & Mid Wind OFF
       rts
L_Wind_Mid:
       bclr 4,LDAT3             ; Max Wind OFF
       mov #%00000111,LDAT6     ; FAN & Min Wind & Mid Wind ON, AUTO OFF
       rts
L_Wind_Auto:
       bclr 4,LDAT3             ; Max Wind OFF
       mov #%00001100,LDAT6     ; FAN & AUTO ON, Min Wind & Mid Wind OFF
       rts

L_Wind_OFF:
       bclr 4,LDAT3
       clr LDAT6                ; All OFF if AC OFF
       rts
; ------------------------------------------------------------------------------ *
LCD_N3_Chk:

       brclr 7,Tx_Data10,L_N3_AC_OFF

L_N3_AC_ON:                     ; Here AC ON
       lda Tx_Data10
       and #%01110000           ; Mask all bit except bit4-6
       cmp #%00000000                                   ; Is bit4-6 = %000
       beq L_N3_Auto_oC         ; Go to Auto mode if true
       bra L_N3_oC_ON

L_N3_Auto_oC:
       clr LDAT7                ; XXoC OFF in AUTO mode
       clr LDAT8
       rts

L_N3_AC_OFF:                    ; Here AC OFF
L_N3_oC_ON:
```

```
        clrh
        lda Tx_Data32
        and #%11110000
        nsa
        tax                     ; Store N3 as Index
        lda $F000,x
        sta LDAT8
        aix #$10
        lda $F000,x
        sta LDAT7
        rts
; ------------------------------------------------------------------------- *
LCD_N45_Chk:

        brset 3,Tx_Data54,L_MODEL_FLASH   ; Flash if Model Set
        bra L_MODEL_ON

L_MODEL_FLASH:

        lda Sleep_Flash_Cnt
        cmp  #$02
        bls  L_MODEL_ON     ; MODEL OFF if ($00-$02)    [125ms*3=0.375s]
        cmp  #$05
        bls  L_MODEL_OFF    ; MODEL ON if ($03-$05)    [0.375s]
        cmp  #$08
        bls  L_MODEL_ON     ; MODEL ON if ($06-$08)    [0.375s]
        cmp  #$0B
        bls  L_MODEL_OFF    ; MODEL ON if ($09-$0B)    [0.375s]
        bra  L_7SEG_CHK

L_MODEL_ON:
        bset 7,LDAT5        ; MODEL flash ON
        bra  L_7SEG_CHK

L_MODEL_OFF:
        bclr 7,LDAT5        ; MODEL flash OFF
        bra  L_7SEG_CHK
; ------------------------- *
L_7SEG_CHK:

        clrh                ; Update 5.3-5.0 7-Segment
        lda Tx_Data54
        nsa                 ; swrap N5 to lower nibble
        and #%00001111      ; mask N4 (Force bit 4-7 =0)
        tax
        lda $F020,x         ; Load 7-segment data
        sta LDAT4           ; Display x(0-9)
; ------------------------- *
        lda Timer_Flash_Cnt
        beq L_Timer_OFF

        rts
; ------------------------------------------------------------------ *
```

```
L_Timer_OFF:

       bset 7,LDAT5                     ; MODEL ON
       rts
; ------------------------------------------------------------------------------ *
LCD_N67_Chk:                           ; Only 6.1 need to check
       brclr 1,Tx_Data10,L_Light_OFF
       bset 0,LDAT3                    ; "light" ON (M.WIND)
       bra L_Light_END
L_Light_OFF:
       bclr 0,LDAT3                    ; "light" OFF (M.WIND)
L_Light_END:

       brclr 7,Tx_Data10,L_N7_AC_OFF   ; Check AC ON/OFF?

L_N7_AC_ON:
       lda Tx_Data10
       and #%01110000                  ; mask off except 1.2, 1.1, 1.0
       beq L_C_OFF

       bra L_N7_AC_OFF

L_C_OFF:
       bclr 4,LDAT8                   ; oC OFF if in AC ON & Auto mode
       bra L_N67_END

L_N7_AC_OFF:
       bset 4,LDAT8                   ; oC ON if AC OFF
L_N67_END:
       rts
; ------------------------------------------------------------------------------ *
; Frame Tx (need to check Tx_Ready flag)
;
; <Need to fine turn the timming of bit transmission>
; ------------------------------------------------------------------------------ *
Tx_Frame:

       brclr TX_READY,Tx_Flag,No_Tx
Tx_S34_Chk:
       brclr S34_KEY_ON,Key_Flag,Tx_Header

No_Tx:
       rts
; ------------------------------------------------------- *
Tx_Header:                     ; Header Code Tx

       SEI                     ; Disable all interrupts before Tx
       bset 1,LDAT9            ; Tx signal ON (A2,A3,A4)
       bset 2,LDAT9
       bset 5,LDAT9
```

```
        bsr T2_ON               ; Send Header High [16us]
        lda #Head_Time_ON       ; 18*0.5ms
        jsr Delay_Hms           ; Delay 9ms

        bsr T2_OFF              ; [16us]
        lda #Head_Time_OFF      ; 9*0.5ms
        jsr Delay_Hms           ; Delay 4.5ms
; ------------------------------------------------------------------------ *
; Tx Data from 2.0 - 9.3
; ------------------------------------------------------------------------ *
        lda Tx_Data10
        bsr Tx_N3_N9
        lda Tx_Data32
        bsr Tx_N3_N9
        lda Tx_Data54
        bsr Tx_N3_N9
        lda Tx_CtmCode
        bsr Tx_N3_N9
        bra Tx_Stop
; ------------------------------------------------------- *
Tx_N3_N9:                       ; Nibble2-3 Tx
        clrh                    ; [1us]
        clrx                    ; [1us]

        sta Tx_Data_Temp
Tx_N3_Loop:
        lsr Tx_Data_Temp        ; [1us]
        blo Data_N3_1           ; [3us] jump if C=1
        bsr Data_0              ; Send Data 0 [4us] if C=0
        bra Tx_N3_Next
Data_N3_1:
        bsr Da                  ; Send Data 1 [4us] if C=1
Tx_N3_Next:
        incx                    ; [1us] b0>x=1 .. b3>x=4
        cpx #$07                ; Is 1.0 - 1.3 complete? Total 4 bits [2us]
        bls Tx_N3_Loop          ; [3us]

        rts
; ------------------------------------------------------- *
Tx_Stop:                      ; Stop bit Tx

        bsr  Data_1             ; Send Data_1 as stop bit !!!

        bsr T2_OFF              ; Stop Timer 2 after STOP bit was sent

        bclr TX_READY,Tx_Flag   ; clear TX_READY to avoid next Tx until other key
                                ; pressed
        clr Tx_Flag

        CLI
        rts
; ------------------------------------------------------- *
```

```
; ------------------------------------------------------------------------ *
; Tx Data "0" OR Data "1"
; ------------------------------------------------------------------------ *
Data_0:
       bsr T2_ON               ; Send Carrier [16us]
       lda #Data0_Time_ON      ; 61*10us + 4us
       jsr Delay_Us            ; Delay 630us - 20us [4us]

       bsr T2_OFF              ; Send Low [16us]
       lda #Data0_Time_OFF     ; 54*10us  {-2 for compensation}
       jsr Delay_Us            ; Delay 560us - 20us
       rts                     ; [4us]

Data_1:

       bsr T2_ON               ; Send Carrier [16us]
       lda #Data1_Time_ON      ; 61*10us
       jsr Delay_Us            ; Delay 630us - 20us

       bsr T2_OFF              ; Send Low [16us]
       lda #Data1_Time_OFF     ; 164*10us {-4 for compensation}
       jsr Delay_Us            ; Delay 1660us - 20us [4us]
       rts                     ; [4us]

; ------------------------------------------------------------------------ *
; T2CH0 ON/OFF Control
; ------------------------------------------------------------------------ *
T2_ON:                         ; Enable TIM2 Now  [16us]
       mov #%00110000,T2SC     ; TOF int disable, Timer stop & reset, clock=bus/1
       mov #%00011010,T2SC0    ; Clear int flag, Int disable,
                               ; clear o/p on compare, toggle o/p on overflow
       bclr 5,T2SC             ; clear TSTOP, enable timer counter

       rts

T2_OFF:                        ; Disable TIM2 Now        [16us]
       mov #%00110000,T2SC     ; TOF int disable, Timer stop & reset, clock=bus/1
       mov #%00010010,T2SC0    ; Clear int flag, Int disable,T2CH0 = port o/p low
       bclr 0,PTB

       rts
```

```
; ------------------------------------------------------------------------- *
; Subroutine <Us Delay>                                                     *
; Bus Clock = 1MHz, 1 Cycle=1uS                                             *
; In      :  Acc                                                            *
; Out     :  <nil>                                                          *
; Call    :  <nil>                                                          *
; Remark  : (10uS * A) Delay                                                *
; ------------------------------------------------------------------------- *
Delay_Us:

Delay_Us_A:
       sta $FFFF               ; [4] clear COP
       nop                     ; [1]
       nop                     ; [1]
       nop                     ; [1]
       dbnza Delay_Us_A        ; [3] 10*{A}

       rts                     ; [4]

;Total= {10AX+[10]}*Bus Cycle
;Total= [4+(10A)]*1uS = 10uS*A + 4uS

; ------------------------------------------------------------------------- *
; Program Interrupt Service Routine Area                                    *
; ------------------------------------------------------------------------- *
;      org       $EE00
; ------------------------------------------------------------------------- *
; DMY_ISR - dummy Interrupt Service Routine (with no operation)             *
; In      :  <nil>                                                          *
; Out     :  <nil>                                                          *
; Call    :  <nil>                                                          *
; ------------------------------------------------------------------------- *
DMY_ISR:
       nop
       rti
; ------------------------------------------------------------------------- *
; Timer Interrupt Service Routine                                           *
; In      :  <nil>                                                          *
; Out     :  <nil>                                                          *
; Call    :  <nil>                                                          *
; ------------------------------------------------------------------------- *
T1M0_ISR:                      ; Unused
       bclr CH0F,T1SC0         ; clear CH0F flag in TIM1
       rti
; ------------------------------------------------------------------------- *
T2M0_ISR:                      ; As TIM2 OCMP
       bclr CH0F,T2SC0         ; clear CH0F flag in TIM2
       rti
; ------------------------------------------------------------------------- *
T1M1_ISR:                      ; Unused
       bclr CH1F,T1SC1         ; clear CH1F flag in TIM1
       rti
```

```
; ------------------------------------------------------------------------ *
T2M1_ISR:                       ; Unused
      bclr CH1F,T2SC1           ; clear CH1F flag in TIM2
      rti
; ------------------------------------------------------------------------ *
T1OF_ISR:                       ; TIM1 Overflow (125ms interrupt period)
       bclr TOF,T1SC            ; clear TIM1 TOF flag in TIM1

       bset LCD_READY,Key_Flag  ; Each interrupt to set LCD READY Flag
                                ; @125ms jsr for LCD

       brset KEY_CONFIRM,Key_Flag,No_T_Flash_Dec
       lda Timer_Flash_Cnt      ; Decrement Timer_Flash_Cnt
       beq T_Time_Out
       deca
       sta Timer_Flash_Cnt
       bra No_T_Flash_Dec

T_Time_Out:
       mov T_Pointer_Tmp,T_Pointer
       mov Tx_Data54_Tmp,Tx_Data54
       bclr 3,Tx_Data54

No_T_Flash_Dec:

       lda Sleep_Flash_Cnt    ; Increment Sleep Flash Counter by each 125ms
       cmp #$0B
       bhs K_Sleep_Clr_Cnt
       inca
       sta Sleep_Flash_Cnt
       bra End_T1OF

K_Sleep_Clr_Cnt:
       clr Sleep_Flash_Cnt

End_T1OF:
       inc Key_Sleep_Cnt      ; Increment sleep counter by each 125ms

       lda S34_Key_Cnt        ; Dec S3 S4 Key pressed timer counter until =0
       beq No_S34_Cnt_Dec
       deca
       sta S34_Key_Cnt

No_S34_Cnt_Dec:

       rti
; ------------------------------------------------------------------------ *
T2OF_ISR:                       ; TIM2 Overflow
      bclr TOF,T2SC             ; clear TIM2 TOF flag in TIM2
      rti
```

```
; ------------------------------------------------------------------------ *
; KBI_ISR - Keyboard & PPI Interrupt Service Routine                       *
; In      :  <nil>                                                         *
; Out     :  <nil>                                                         *
; Call    :  <nil>                                                         *
; Remark  : Either PPI or KBI enable at the same time                      *
; ------------------------------------------------------------------------ *
KBI_ISR:

     lda PTB                    ; Toggles PTB1 when KBI interrupt present
     eor #%00000010             ; For Debug
     sta PTB

PPI_CHK:
     brset PPI1L,HDB,PPI_ACK    ; check PPI interrupt flag
     bra KBI_ACK                ; If not a PPI interrupt, jump to KBI_ACK

PPI_ACK:
     bset ACKK,KBSCR            ; clear PPI1L flag by set ACKK bit
     brset PPI1L,HDB,PPI_ACK    ; check PPI int flag clear

     bset 6,PPI_Cnt
     rti

KBI_ACK:
     brclr 6,PPI_Cnt,KBI_Normal
     bset 7,PPI_Cnt
KBI_Normal:
     bset KEY_ON,Key_Flag       ; Set KEY_ON flag

     bset ACKK,KBSCR            ; clear PPI1L flag by set ACKK bit
     brset KEYF,KBSCR,KBI_Normal
     rti
; ------------------------------------------------------------------------ *
; LVI_ISR - LVI Interrupt Service Routine                                  *
; In      :  <nil>                                                         *
; Out     :  <nil>                                                         *
; Call    :  <nil>                                                         *
; ------------------------------------------------------------------------ *
LVI_ISR:
       lda #%01010000
       sta LVISR              ; clear LVI int. flag (Cannot use bset 4,LVISR)
       rti                    ; Cannot use "bset 4,LVISR" due to LVISR = $FE0F)
; ------------------------------------------------------------------------ *
; IRQ_ISR - IRQ Interrupt Service Routine                                  *
; In      :  <nil>                                                         *
; Out     :  <nil>                                                         *
; Call    :  <nil>                                                         *
; ------------------------------------------------------------------------ *
IRQ_ISR:
       bset 2,INTSCR          ; CLEAR IRQF
       rti
```

```
; ------------------------------------------------------------------------ *
; Program Look Up Table Area (For LCD #1)                                  *
; ------------------------------------------------------------------------ *
      org       $F000         ;$F000-$F01F
; ------------------------------------------------------------------------ *
; LCD oC Display Lookup Table

;           ABC%XGED   ; 10th digit of oC (%=oC & TEMP always ON, X always OFF)
      FCB %01110000   ; Digit 1 (B,C)         [X=$00]
      FCB %01110000   ; Digit 1 (B,C)         [X=$01]
      FCB %01110000   ; Digit 1 (B,C)         [X=$02]
      FCB %01110000   ; Digit 1 (B,C)         [X=$03]
      FCB %01110000   ; Digit 1 (B,C)         [X=$04]

      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$05]
      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$06]
      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$07]
      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$08]
      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$09]
      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$0A]
      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$0B]
      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$0C]
      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$0D]
      FCB %11010111   ; Digit 2 (A,B,D,E,G)   [X=$0E]

      FCB %11110101   ; Digit 3 (A,B,C,D,G)   [X=$0F]
; ------------------------------------------------------------------- *
;           ABCXFGED   ; 1st digit of oC (X always OFF)
      FCB %10101101   ; Digit 5 (A,C,D,F,G)      [X=$00+$10]
      FCB %10101111   ; Digit 6 (A,C,D,E,F,G)    [X=$01+$10]
      FCB %11100000   ; Digit 7 (A,B,C)          [X=$02+$10]
      FCB %11101111   ; Digit 8 (A,B,C,D,E,F,G)  [X=$03+$10]
      FCB %11101101   ; Digit 9 (A,B,C,D,F,G)    [X=$04+$10]

      FCB %11101011   ; Digit 0 (A,B,C,D,E,F)    [X=$05+$10]
      FCB %01100000   ; Digit 1 (B,C)            [X=$06+$10]
      FCB %11000111   ; Digit 2 (A,B,D,E,G)      [X=$07+$10]
      FCB %11100101   ; Digit 3 (A,B,C,D,G)      [X=$08+$10]
      FCB %01101100   ; Digit 4 (B,C,F,G)        [X=$09+$10]
      FCB %10101101   ; Digit 5 (A,C,D,F,G)      [X=$0A+$10]
      FCB %10101111   ; Digit 6 (A,C,D,E,F,G)    [X=$0B+$10]
      FCB %11100000   ; Digit 7 (A,B,C)          [X=$0C+$10]
      FCB %11101111   ; Digit 8 (A,B,C,D,E,F,G)  [X=$0D+$10]
      FCB %11101101   ; Digit 9 (A,B,C,D,F,G)    [X=$0E+$10]

      FCB %11101011   ; Digit 0 (A,B,C,D,E,F)    [X=$0F+$10]
```

```
; ------------------------------------------------------------------------ *
      org       $F020         ;$F020-$F029
; ------------------------------------------------------------------------ *
; LCD Model Setting Display Lookup Table

;           ABCXFGED   ;
      FCB %11101011   ; Digit 0 (A,B,C,D,E,F)    [X=$00]
      FCB %01100000   ; Digit 1 (B,C)            [X=$01]
      FCB %11000111   ; Digit 2 (A,B,D,E,G)      [X=$02]
      FCB %11100101   ; Digit 3 (A,B,C,D,G)      [X=$03]
      FCB %01101100   ; Digit 4 (B,C,F,G)        [X=$04]
      FCB %10101101   ; Digit 5 (A,C,D,F,G)      [X=$05]
      FCB %10101111   ; Digit 6 (A,C,D,E,F,G)    [X=$06]
      FCB %11100000   ; Digit 7 (A,B,C)          [X=$07]
      FCB %11101111   ; Digit 8 (A,B,C,D,E,F,G)  [X=$08]
      FCB %11101101   ; Digit 9 (A,B,C,D,F,G)    [X=$09]

; ------------------------------------------------------------------------ *
      org       $F030         ;$F030-$F039
; ------------------------------------------------------------------------ *
; Key Press Model Setting Lookup Table

; from Model 0(x=00) to Model 9 (X=09);
; ------------------------------------------- *

      FCB %00001000   ; 0 Model     [X=$00]
      FCB %00011000   ; 1 Model     [X=$01]
      FCB %00101000   ; 2 Model     [X=$02]
      FCB %00111000   ; 3 Model     [X=$03]
      FCB %01001000   ; 4 Model     [X=$04]
      FCB %01011000   ; 5 Model     [X=$05]
      FCB %01101000   ; 6 Model     [X=$06]
      FCB %01111000   ; 7 Model     [X=$07]
      FCB %10001000   ; 8 Model     [X=$08]
      FCB %10011000   ; 9 Model     [X=$09]

; ------------------------------------------------------------------------ *
```

B.2 Receiver Listing

```
; ------------------------------------------------------------------------------ *
; Freescale Semiconductor (H.K.) Ltd.                                            *
; 8/16 bit MCU - Application                                                     *
;                                                                                *
; FileName      :  Main.asm (Receiver)                                           *
; Title         :  08LT8 Remote Control Reference Demo code (Freescale)          *
; MCU           :  PC68HC908LT8CFB (44-LQFP) 1st Silicon (Mask Set 0M48C) only   *
; Assembler     :  Metrowerks CodeWarrior HC(S)08 (v3.1)                         *
; Include File  :  908LT8v0r0.inc (for LT8 1st silicon only)                     *
; Author        :  T.C. Lun                                                      *
;                                                                                *
;                  DD/MM/YY     Rev.    Modified comments                        *
; History       :  15/12/05     0.0     Initial release                          *
;                                                                                *
; Introduction  : The H/W setting are show as below:                             *
;    PTA[1:0]   : KBI of Keys, pullup if enable                                  *
;    PTA[2-7]   : NC set o/p low                                                 *
;    PTB0/T2CH0 : IR Tx Module (38.46KHz Demodulator)                            *
;    PTB[1]     : NC set o/p low                                                 *
;    PTB[2]     : Red LED for AC power OFF (high current pin)                    *
;    PTB[3]     : Green LED for AC power ON (high current pin)                   *
;    BP0-3      : Connect to LCD Panel Pin4-1                                    *
;    FP4-10     : Connect to LCD Panel Pin5-11                                   *
;    FP12-18    : Connect to LCD Panel Pin12-18                                  *
;    FP3,11 & FP19-22 : NC o/p                                                   *
;    RST        : 10K pullup + 0.1uF to GND                                      *
;    IRQ        : NC connect 0.1uF to GND                                        *
;    OSC1 & 2   : 4M7 + 4MHz + 18pF x2 OR (1M + 4MHz Resonator)                  *
;    XTAL1 & 2  : 10M + 10K + 32.768KHz + 10pF x2                                *
;                 Vdd & Vss : 0.01uF // 10uF                                     *
;    Unused pin : PTB6-7 & PTC4-7 (need to set as o/p low) available in 52-LQFP  *
;                                                                                *
;  Special arrangment : Add 100uF near to Vdd of Tx diode                        *
;    : Add Mon08 interface for programming/ICD                                   *
;                                                                                *
; MON08 Interface:                                                               *
;  PIN# Net Name    | PIN# Net Name                                              *
;  1  NC            |  2   GND                                                   *
;  3  NC            |  4   /RST                                                  *
;  5  NC            |  6   /IRQ(+100 ohm)                                        *
;  7  NC            |  8   PTA0                                                  *
;  9  NC            |  10  NC                                                    *
;  11 NC            |  12  PTA1                                                  *
;  13 OSC1(+22 Ohm) |  14  PTA2                                                  *
;  15 Vdd           |  16  PTC3                                                  *
;                                                                                *
;                                                                                *
; Remark : This code is only for 1st 908LT8 silicon                              *
;          For new silicon, it need to change the PPI interrupt vector           *
;          and its register & handling !!!!                                      *
;                                                                                *
;                                                                                *
```

```
; ------------------------------------------------------------------------- *
; Disclaimer of All Warranties & Liabilities :                              *
;  This Program is a freeware to demonstrate the operation of HC08 micro-   *
;  controller. In no event will Motorola be liable for any damages, or any  *
;  incidental or consequential damages arising out of the use of or         *
;  inability to use this program. User agrees that Motorola does not make   *
;  any warranties of any kind that the program does not or will not         *
;  infringe any copyright, patent, trade secret or other intellectual       *
;  property right of any third party in any country.                        *
; ------------------------------------------------------------------------- *

            XDEF Entry, IRQ_ISR, main, LVI_ISR, T1M0_ISR, T2M0_ISR, T1M1_ISR,
            T2M1_ISR, T1OF_ISR, T2OF_ISR, KBI_ISR, DMY_ISR
            Include '908LT8v0r0.inc'
;*****************************************************************************
;* Title: 908LV8.inc   (c) Freescale Semiconductor, Inc. 2004 All rights reserved
;*****************************************************************************
;* Author: T.C. Lun - Freescale TSPG
;*
;* Description: Register and bit name definitions for MC68HC908LT8
;*
;* Documentation:  MC68HC908LT8/D
;*
;* Include Files: none
;*
;* Assembler:  Metrowerks Code Warrior 3.0
;*             or P&E Microcomputer Systems - CASM08Z (v. 3.16)
;*
;* Revision History:
;* Rev #     Date      Who     Comments
;* -----  -----------  ------  ----------------------------------------------
;*  0.0    10/14/04    TC.Lun   (Rev.0.3 Preliminary Draft is used)
;*
;*****************************************************************************
;****  Memory Map and Interrupt Vectors  ************************************
;*
RamStart:   equ   $0080       ;start of RAM
RamLast:    equ   $00FF       ;last RAM location
RomStart:   equ   $DE00       ;start of Flash
RomLast:    equ   $FFFF       ;last Flash location
MonStart:   equ   $0B97       ;start of monitor ROM

Vkbd:       equ   $FFDC       ;keyboard vector
Vtim2ov:    equ   $FFEA       ;timer 2 overflow
Vtim2ch1:   equ   $FFEC       ;timer 2 channel 1 vector
Vtim2ch0:   equ   $FFEE       ;timer 2 channel 0 vector
Vtimov:     equ   $FFF0       ;timer 1 overflow vector
Vtimch1:    equ   $FFF2       ;timer 1 channel 1 vector
Vtimch0:    equ   $FFF4       ;timer 1 channel 0 vector
Vlvi:       equ   $FFF8       ;LVI interrupt vector
Virq:       equ   $FFFA       ;IRQ vector
Vswi:       equ   $FFFC       ;SWI vector
Vreset:     equ   $FFFE       ;reset vector
```

```
;****  Input/Output (I/O) Ports  ************************************************
;*
PTA:        equ   $00         ;port A data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTA7:       equ   7           ;port A data bit 7
PTA6:       equ   6           ;port A data bit 6
PTA5:       equ   5           ;port A data bit 5
PTA4:       equ   4           ;port A data bit 4
PTA3:       equ   3           ;port A data bit 3
PTA2:       equ   2           ;port A data bit 2
PTA1:       equ   1           ;port A data bit 1
PTA0:       equ   0           ;port A data bit 0
; bit position masks
mPTA7:      equ   %10000000   ;port A data bit 7
mPTA6:      equ   %01000000   ;port A data bit 6
mPTA5:      equ   %00100000   ;port A data bit 5
mPTA4:      equ   %00010000   ;port A data bit 4
mPTA3:      equ   %00001000   ;port A data bit 3
mPTA2:      equ   %00000100   ;port A data bit 2
mPTA1:      equ   %00000010   ;port A data bit 1
mPTA0:      equ   %00000001   ;port A data bit 0

PTB:        equ   $01         ;port B data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTB7:       equ   7           ;port B data bit 7
PTB6:       equ   6           ;port B data bit 6
PTB3:       equ   3           ;port B data bit 3
PTB2:       equ   2           ;port B data bit 2
PTB1:       equ   1           ;port B data bit 1
PTB0:       equ   0           ;port B data bit 0
; bit position masks
mPTB7:      equ   %10000000   ;port B data bit 7
mPTB6:      equ   %01000000   ;port B data bit 6
mPTB3:      equ   %00001000   ;port B data bit 3
mPTB2:      equ   %00000100   ;port B data bit 2
mPTB1:      equ   %00000010   ;port B data bit 1
mPTB0:      equ   %00000001   ;port B data bit 0

PTC:        equ   $02         ;port C data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTC7:       equ   7           ;port C data bit 7
PTC6:       equ   6           ;port C data bit 6
PTC5:       equ   5           ;port C data bit 5
PTC4:       equ   4           ;port C data bit 4
PTC3:       equ   3           ;port C data bit 3
PTC2:       equ   2           ;port C data bit 2
PTC1:       equ   1           ;port C data bit 1
PTC0:       equ   0           ;port C data bit 0
; bit position masks
mPTC7:      equ   %10000000   ;port C data bit 7
mPTC6:      equ   %01000000   ;port C data bit 6
mPTC5:      equ   %00100000   ;port C data bit 5
mPTC4:      equ   %00010000   ;port C data bit 4
mPTC3:      equ   %00001000   ;port C data bit 3
```

```
mPTC2:      equ   %00000100   ;port C data bit 2
mPTC1:      equ   %00000010   ;port C data bit 1
mPTC0:      equ   %00000001   ;port C data bit 0

PTD:        equ   $03         ;port D data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTD7:       equ   7           ;port D data bit 7
PTD6:       equ   6           ;port D data bit 6
PTD5:       equ   5           ;port D data bit 5
PTD4:       equ   4           ;port D data bit 4
PTD3:       equ   3           ;port D data bit 3
PTD2:       equ   2           ;port D data bit 2
PTD1:       equ   1           ;port D data bit 1
PTD0:       equ   0           ;port D data bit 0
; bit position masks
mPTD7:      equ   %10000000   ;port D data bit 7
mPTD6:      equ   %01000000   ;port D data bit 6
mPTD5:      equ   %00100000   ;port D data bit 5
mPTD4:      equ   %00010000   ;port D data bit 4
mPTD3:      equ   %00001000   ;port D data bit 3
mPTD2:      equ   %00000100   ;port D data bit 2
mPTD1:      equ   %00000010   ;port D data bit 1
mPTD0:      equ   %00000001   ;port D data bit 0

DDRA:       equ   $04         ;port A data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRA7:      equ   7           ;port A data direction bit 7
DDRA6:      equ   6           ;port A data direction bit 6
DDRA5:      equ   5           ;port A data direction bit 5
DDRA4:      equ   4           ;port A data direction bit 4
DDRA3:      equ   3           ;port A data direction bit 3
DDRA2:      equ   2           ;port A data direction bit 2

DDRA1:      equ   1           ;port A data direction bit 1
DDRA0:      equ   0           ;port A data direction bit 0
; bit position masks
mDDRA7:     equ   %10000000   ;port A data direction bit 7
mDDRA6:     equ   %01000000   ;port A data direction bit 6
mDDRA5:     equ   %00100000   ;port A data direction bit 5
mDDRA4:     equ   %00010000   ;port A data direction bit 4
mDDRA3:     equ   %00001000   ;port A data direction bit 3
mDDRA2:     equ   %00000100   ;port A data direction bit 2
mDDRA1:     equ   %00000010   ;port A data direction bit 1
mDDRA0:     equ   %00000001   ;port A data direction bit 0

DDRB:       equ   $05         ;port B data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRB7:      equ   7           ;port B data direction bit 7
DDRB6:      equ   6           ;port B data direction bit 6
DDRB3:      equ   3           ;port B data direction bit 3
DDRB2:      equ   2           ;port B data direction bit 2
DDRB1:      equ   1           ;port B data direction bit 1
DDRB0:      equ   0           ;port B data direction bit 0
```

```
; bit position masks
mDDRB7:     equ   %10000000   ;port B data direction bit 7
mDDRB6:     equ   %01000000   ;port B data direction bit 6
mDDRB3:     equ   %00001000   ;port B data direction bit 3
mDDRB2:     equ   %00000100   ;port B data direction bit 2
mDDRB1:     equ   %00000010   ;port B data direction bit 1
mDDRB0:     equ   %00000001   ;port B data direction bit 0

DDRC:       equ   $06         ;port C data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRC7:      equ   7           ;port C data direction bit 7
DDRC6:      equ   6           ;port C data direction bit 6
DDRC5:      equ   5           ;port C data direction bit 5
DDRC4:      equ   4           ;port C data direction bit 4
DDRC3:      equ   3           ;port C data direction bit 3
DDRC2:      equ   2           ;port C data direction bit 2
DDRC1:      equ   1           ;port C data direction bit 1
DDRC0:      equ   0           ;port C data direction bit 0
; bit position masks
mDDRC7:     equ   %10000000   ;port C data direction bit 7
mDDRC6:     equ   %01000000   ;port C data direction bit 6
mDDRC5:     equ   %00100000   ;port C data direction bit 5
mDDRC4:     equ   %00010000   ;port C data direction bit 4
mDDRC3:     equ   %00001000   ;port C data direction bit 3
mDDRC2:     equ   %00000100   ;port C data direction bit 2
mDDRC1:     equ   %00000010   ;port C data direction bit 1
mDDRC0:     equ   %00000001   ;port C data direction bit 0

DDRD:       equ   $07         ;port D data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRD7:      equ   7           ;port D data direction bit 7
DDRD6:      equ   6           ;port D data direction bit 6
DDRD5:      equ   5           ;port D data direction bit 5
DDRD4:      equ   4           ;port D data direction bit 4
DDRD3:      equ   3           ;port D data direction bit 3
DDRD2:      equ   2           ;port D data direction bit 2
DDRD1:      equ   1           ;port D data direction bit 1
DDRD0:      equ   0           ;port D data direction bit 0
; bit position masks
mDDRD7:     equ   %10000000   ;port D data direction bit 7
mDDRD6:     equ   %01000000   ;port D data direction bit 6
mDDRD5:     equ   %00100000   ;port D data direction bit 5
mDDRD4:     equ   %00010000   ;port D data direction bit 4
mDDRD3:     equ   %00001000   ;port D data direction bit 3
mDDRD2:     equ   %00000100   ;port D data direction bit 2
mDDRD1:     equ   %00000010   ;port D data direction bit 1
mDDRD0:     equ   %00000001   ;port D data direction bit 0

DDRE:       equ   $08         ;port E data direction register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
DDRE7:      equ   7           ;port E data direction bit 7
DDRE6:      equ   6           ;port E data direction bit 6
DDRE5:      equ   5           ;port E data direction bit 5
DDRE4:      equ   4           ;port E data direction bit 4
```

```
DDRE3:      equ   3           ;port E data direction bit 3
DDRE2:      equ   2           ;port E data direction bit 2
DDRE1:      equ   1           ;port E data direction bit 1
DDRE0:      equ   0           ;port E data direction bit 0
; bit position masks
mDDRE7:     equ   %10000000   ;port E data direction bit 7
mDDRE6:     equ   %01000000   ;port E data direction bit 6
mDDRE5:     equ   %00100000   ;port E data direction bit 5
mDDRE4:     equ   %00010000   ;port E data direction bit 4
mDDRE3:     equ   %00001000   ;port E data direction bit 3
mDDRE2:     equ   %00000100   ;port E data direction bit 2
mDDRE1:     equ   %00000010   ;port E data direction bit 1
mDDRE0:     equ   %00000001   ;port E data direction bit 0

PTE:        equ   $09         ;port E data register
; bit numbers for use in BLCR, BSET, BRCLR, and BRSET
PTE7:       equ   7           ;port E data bit 7
PTE6:       equ   6           ;port E data bit 6
PTE5:       equ   5           ;port E data bit 5
PTE4:       equ   4           ;port E data bit 4
PTE3:       equ   3           ;port E data bit 3
PTE2:       equ   2           ;port E data bit 2
PTE1:       equ   1           ;port E data bit 1
PTE0:       equ   0           ;port E data bit 0
; bit position masks
mPTE7:      equ   %10000000   ;port E data bit 7
mPTE6:      equ   %01000000   ;port E data bit 6
mPTE5:      equ   %00100000   ;port E data bit 5
mPTE4:      equ   %00010000   ;port E data bit 4
mPTE3:      equ   %00001000   ;port E data bit 3
mPTE2:      equ   %00000100   ;port E data bit 2
mPTE1:      equ   %00000010   ;port E data bit 1
mPTE0:      equ   %00000001   ;port E data bit 0

HDB:        equ   $0C         ;port B high current drive control
                              ;register
; bit number for use in BCLR, BSET, BRCLR, BRSET
PPI1L:      equ   6           ;PPI1 interrupt request level
HDB3:       equ   3           ;port B3 high current drive enable
HDB2:       equ   2           ;port B2 high current drive enable
PPI1CLKS1:  equ   1           ;PPI1 clock select 1
PPI1CLKS0:  equ   0           ;PPI1 clock select 0

mPPI1L:     equ   %01000000   ;PPI1 interrupt request level
mHDB3:      equ   %00001000   ;port B3 high current drive enable
mHDB2:      equ   %00000100   ;port B2 high current drive enable
mPPI1CLKS1: equ   %00000010   ;PPI1 clock select 1
mPPI1CLKS0: equ   %00000001   ;PPI1 clock select 0

;****  Keyboard Interrupt Module (KBI)  ************************************
;*
KBSCR:      equ   $1B         ;keyboard status and control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
KEYF:       equ   3           ;keyboard flag
```

```
ACKK:       equ   2           ;keyboard acknowledge
IMASKK:     equ   1           ;keyboard interrupt mask
MODEK:      equ   0           ;keyboard triggering sesitivity
; bit position masks
mKEYF:      equ   %00001000   ;keyboard flag
mACKK:      equ   %00000100   ;keyboard acknowledge
mIMASKK:    equ   %00000010   ;keyboard interrupt mask
mMODEK:     equ   %00000001   ;keyboard triggering sesitivity

KBIER:      equ   $1C         ;keyboard interrupt enable register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
PPI1IE2:    equ   6           ;PPI1 interrupt enable and frequency
                              ;select bit 2
PPI1IE1:    equ   5           ;PPI1 interrupt enable and frequency
                              ;select bit 1
PPI1IE0:    equ   4           ;PPI1 interrupt enable and frequency
                              ;select bit 0
KBIE3:      equ   3           ;port A keyboard interrupt enable bit 3
KBIE2:      equ   2           ;port A keyboard interrupt enable bit 2
KBIE1:      equ   1           ;port A keyboard interrupt enable bit 1
KBIE0:      equ   0           ;port A keyboard interrupt enable bit 0
; bit position masks
mPPI1IE2:   equ   %01000000   ;PPI1 interrupt enable and frequency
                              ;select bit 2
mPPI1IE1:   equ   %00100000   ;PPI1 interrupt enable and frequency
                              ;select bit 1
mPPI1IE0:   equ   %00010000   ;PPI1 interrupt enable and frequency
                              ;select bit 0
mKBIE3:     equ   %00001000   ;port A keyboard interrupt enable bit 3
mKBIE2:     equ   %00000100   ;port A keyboard interrupt enable bit 2
mKBIE1:     equ   %00000010   ;port A keyboard interrupt enable bit 1
mKBIE0:     equ   %00000001   ;port A keyboard interrupt enable bit 0

;**** Configuration Registers 2 (CONFIG2) ************************************
;*
CONFIG2:    equ   $1D         ;configuration register 2
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
STOP_XCLKEN: equ   7          ;stop osc1 & osc2 crystal clock enable
STOP_XTALEN: equ   6          ;stop xtal1 & xtal2 crystal clock enable
PEE:        equ   5           ;port E LCD or GPIO select
PDE:        equ   4           ;port D LCD or GPIO select
PCEH:       equ   3           ;port C higher nibble lcd or GPIO select
PCEL:       equ   2           ;port C lower nibble lcd or GPIO select
LVISEL1:    equ   1           ;LVI level select bit 1
LVISEL0:    equ   0           ;LVI level select bit 0
; bit position masks
mSTOP_XCLKEN: equ   %10000000 ;stop osc1 & osc2 crystal clock enable
mSTOP_XTALEN: equ   %01000000 ;stop xtal1 & xtal2 crystal clock enable
mPEE:       equ   %00100000   ;port E LCD or GPIO select
mPDE:       equ   %00010000   ;port D LCD or GPIO select
mPCEH:      equ   %00001000   ;port C higher nibble lcd or GPIO select
mPCEL:      equ   %00000100   ;port C lower nibble lcd or GPIO select
mLVISEL1:   equ   %00000010   ;LVI level select bit 1
mLVISEL0:   equ   %00000001   ;LVI level select bit 0
```

```
;****  External Interrupt (IRQ)  ***********************************************
;*
INTSCR:     equ   $1E         ;IRQ status and control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
IRQF:       equ   3           ;IRQ flag
ACK:        equ   2           ;IRQ interrupt request acknowledge
IMASK:      equ   1           ;IRQ interrupt mask
MODE:       equ   0           ;IRQ edge/level select
; bit position masks
mIRQF:      equ   %00001000   ;IRQ flag
mACK:       equ   %00000100   ;IRQ interrupt request acknowledge
mIMASK:     equ   %00000010   ;IRQ interrupt mask
mMODE:      equ   %00000001   ;IRQ edge/level select

;**** Configuration Register 1 (CONFIG1) ***************************************
;*
CONFIG1:    equ   $1F         ;configuration register 1
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
COPRS:      equ   7           ;COP reset period selection
LVISTOP:    equ   6           ;LVI enable in stop mode
LVIRSTD:    equ   5           ;LVI reset disable
LVIPWRD:    equ   4           ;LVI power disable
SSREC:      equ   2           ;short stop recovery
STOP:       equ   1           ;STOP instruction enable
COPD:       equ   0           ;COP disable
; bit position masks
mCOPRS:     equ   %10000000   ;COP reset period selection
mLVISTOP:   equ   %01000000   ;LVI enable in stop mode
mLVIRSTD:   equ   %00100000   ;LVI reset disable
mLVIPWRD:   equ   %00010000   ;LVI power disable
mSSREC:     equ   %00000100   ;short stop recovery
mSTOP:      equ   %00000010   ;STOP instruction enable
mCOPD:      equ   %00000001   ;COP disable

;****  Timer Interface module 1 (TIM1)  ***************************************
T1SC:       equ   $20         ;timer 1 status and control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
TOF:        equ   7           ;TIM overflow flag
TOIE:       equ   6           ;TIM overflow interrupt enable
TSTOP:      equ   5           ;TIM stop bit
TRST:       equ   4           ;TIM reset bit
PS2:        equ   2           ;prescaler select bit 2
PS1:        equ   1           ;prescaler select bit 1
PS0:        equ   0           ;prescaler select bit 0
; bit position masks
mTOF:       equ   %10000000   ;TIM overflow flag
mTOIE:      equ   %01000000   ;TIM overflow interrupt enable
mTSTOP:     equ   %00100000   ;TIM stop bit
mTRST:      equ   %00010000   ;TIM reset bit
mPS2:       equ   %00000100   ;prescaler select bit 2
mPS1:       equ   %00000010   ;prescaler select bit 1
mPS0:       equ   %00000001   ;prescaler select bit 0
```

```
T1SC0:      equ   $25         ;timer 1 channel 0 status and control
                              ;register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
CH0F:       equ   7           ;channel 0 flag
CH0IE:      equ   6           ;channel 0 interrupt enable
MS0B:       equ   5           ;mode select bit B
MS0A:       equ   4           ;mode select bit A
ELS0B:      equ   3           ;edge/level select bit B
ELS0A:      equ   2           ;edge/level select bit A
TOV0        equ   1           ;toggle on overflow
CH0MAX      equ   0           ;channel 0 maximum duty cycle
; bit position masks
mCH0F:      equ   %10000000   ;channel 0 flag
mCH0IE:     equ   %01000000   ;channel 0 interrupt enable
mMS0B:      equ   %00100000   ;mode select bit B
mMS0A:      equ   %00010000   ;mode select bit A
mELS0B:     equ   %00001000   ;edge/level select bit B
mELS0A:     equ   %00000100   ;edge/level select bit A
mTOV0       equ   %00000010   ;toggle on overflow
mCH0MAX     equ   %00000001   ;channel 0 maximum duty cycle

T1SC1:      equ   $28         ;timer 1 channel 1 status and control
                              ;register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
CH1F:       equ   7           ;channel 1 flag
CH1IE:      equ   6           ;channel 1 interrupt enable
MS1A:       equ   4           ;mode select bit A
ELS1B:      equ   3           ;edge/level select bit B
ELS1A:      equ   2           ;edge/level select bit A
TOV1        equ   1           ;toggle on overflow
CH1MAX      equ   0           ;channel 1 maximum duty cycle
; bit position masks
mCH1F:      equ   %10000000   ;channel 1 flag
mCH1IE:     equ   %01000000   ;channel 1 interrupt enable
mMS1A:      equ   %00010000   ;mode select bit A
mELS1B:     equ   %00001000   ;edge/level select bit B
mELS1A:     equ   %00000100   ;edge/level select bit A
mTOV1       equ   %00000010   ;toggle on overflow
mCH1MAX     equ   %00000001   ;channel 1 maximum duty cycle

T1CNTH:     equ   $21         ;timer 1 counter register high
T1CNTL:     equ   $22         ;timer 1 counter register Low
T1MODH:     equ   $23         ;timer 1 counter modulo register high
T1MODL:     equ   $24         ;timer 1 counter modulo register low
T1CH0H:     equ   $26         ;timer 1 channel 0 register high
T1CH0L:     equ   $27         ;timer 1 channel 0 register low
T1CH1H:     equ   $29         ;timer 1 channel 1 register high
T1CH1L:     equ   $2A         ;timer 1 channel 1 register low

;****  Timer Interface module 2 (TIM2)  ************************************
T2SC:       equ   $2B         ;timer 2 status and control register
T2CNTH:     equ   $2C         ;timer 2 counter register high
T2CNTL:     equ   $2D         ;timer 2 counter register low
T2MODH:     equ   $2E         ;timer 2 counter modulo register high
```

```
T2MODL:     equ   $2F         ;timer 2 counter modulo register low
T2SC0:      equ   $30         ;timer 2 channel 0 status and control
                              ;register
T2CH0H:     equ   $31         ;timer 2 channel 0 register high
T2CH0L:     equ   $32         ;timer 2 channel 0 register low
T2SC1:      equ   $33         ;timer 2 channel 1 status and control
                              ;register
T2CH1H:     equ   $34         ;timer 2 channel 1 register high
T2CH1L:     equ   $35         ;timer 2 channel 1 register low

;****  LCD Driver  ************************************************************
LCDCLK:     equ   $4F         ;LCD clock register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
FCCTL1:     equ   6           ;fast charge duty cycle select 1
FCCTL0:     equ   5           ;fast charge duty cycle select 0
DUTY1:      equ   4           ;duty cycle select 1
DUTY0:      equ   3           ;duty cycle select 0
LCLK2:      equ   2           ;LCD clock select bit 2
LCLK1:      equ   1           ;LCD clock select bit 1
LCLK0:      equ   0           ;LCD clock select bit 0
; bit position masks
mFCCTL1:    equ   %01000000   ;fast charge duty cycle select 1
mFCCTL0:    equ   %00100000   ;fast charge duty cycle select 0
mDUTY1:     equ   %00010000   ;duty cycle select 1
mDUTY0:     equ   %00001000   ;duty cycle select 0
mLCLK2:     equ   %00000100   ;LCD clock select bit 2
mLCLK1:     equ   %00000010   ;LCD clock select bit 1
mLCLK0:     equ   %00000001   ;LCD clock select bit 0

LCDCR:      equ   $51         ;LCD control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
LCDE:       equ   7           ;LCD enable
FC:         equ   5           ;fast charge
LC:         equ   4           ;Low current
LCCON3:     equ   3           ;LCD contrast control bit3
LCCON2:     equ   2           ;LCD contrast control bit2
LCCON1:     equ   1           ;LCD contrast control bit1
LCCON0:     equ   0           ;LCD contrast control bit0
; bit position masks
mLCDE:      equ   %10000000   ;LCD enable
mFC:        equ   %00100000   ;fast charge
mLC:        equ   %00010000   ;Low current
mLCCON3:    equ   %00001000   ;LCD contrast control bit3
mLCCON2:    equ   %00000100   ;LCD contrast control bit2
mLCCON1:    equ   %00000010   ;LCD contrast control bit1
mLCCON0:    equ   %00000001   ;LCD contrast control bit0

LDAT1:      equ   $52         ;LCD display data register 1
LDAT2:      equ   $53         ;LCD display data register 2
LDAT3:      equ   $54         ;LCD display data register 3
LDAT4:      equ   $55         ;LCD display data register 4
LDAT5:      equ   $56         ;LCD display data register 5
LDAT6:      equ   $57         ;LCD display data register 6
LDAT7:      equ   $58         ;LCD display data register 7
```

```
LDAT8:      equ   $59         ;LCD display data register 8
LDAT9:      equ   $5A         ;LCD display data register 9
LDAT10:     equ   $5B         ;LCD display data register 10
LDAT11:     equ   $5C         ;LCD display data register 11
LDAT12:     equ   $5D         ;LCD display data register 12
LDAT13:     equ   $5E         ;LCD display data register 13

;****  System Integration Module (SIM)  ************************************
SBSR:       equ   $FE00       ;SIM break status register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
SBSW        equ   1           ;SIM break stop/wait
; bit position masks
mSBSW:      equ   %00000010   ;SIM break stop/wait

SRSR:       equ   $FE01       ;SIM reset status register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
POR:        equ   7           ;power-on reset
PIN:        equ   6           ;external reset
COP:        equ   5           ;COP reset
ILOP:       equ   4           ;illegal opcode reset
ILAD:       equ   3           ;illegal address reset
MODRST:     equ   2           ;monitor mode entry module reset
LVI:        equ   1           ;LVI reset
; bit position masks
mPOR:       equ   %10000000   ;power-on reset
mPIN:       equ   %01000000   ;external reset
mCOP:       equ   %00100000   ;COP reset
mILOP:      equ   %00010000   ;illegal opcode reset
mILAD:      equ   %00001000   ;illegal address reset
mMODRST:    equ   %00000100   ;monitor mode entry module reset
mLVI:       equ   %00000010   ;LVI reset

SBFCR:      equ   $FE03       ;SIM break flag control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
BCFE:       equ   7           ;break clear flag enable
; bit position masks
mBCFE:      equ   %10000000   ;break clear flag enable

INT1:       equ   $FE04       ;interrupt status register 1
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
IF6:        equ   7           ;interrupt flag 6
IF5:        equ   6           ;interrupt flag 5
IF4:        equ   5           ;interrupt flag 4
IF3:        equ   4           ;interrupt flag 3
IF2:        equ   3           ;interrupt flag 2
IF1:        equ   2           ;interrupt flag 1

INT2:       equ   $FE05       ;interrupt status register 2
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
IF9:        equ   2           ;interrupt flag 9
IF8:        equ   1           ;interrupt flag 8
IF7:        equ   0           ;interrupt flag 7
```

```
INT3:       equ   $FE06       ;interrupt status register 3
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
IF17:       equ   2           ;interrupt flag 17
IF16:       equ   1           ;interrupt flag 16

;****  Flash Memory  ********************************************************
FLCR:       equ   $FE08       ;flash control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
HVEN:       equ   3           ;high-voltage enable bit mask
MASS:       equ   2           ;mass erase control bit mask
ERASE:      equ   1           ;erase control bit mask
PGM:        equ   0           ;program control bit mask
; bit position masks
mHVEN:      equ   %00001000   ;high-voltage enable bit mask
mMASS:      equ   %00000100   ;mass erase control bit mask
mERASE:     equ   %00000010   ;erase control bit mask
mPGM:       equ   %00000001   ;program control bit mask

FLBPR:      equ   $FF7E       ;flash block protect register

;****  Break Module (BRK)  **************************************************
BRKSCR:     equ   $FE0E       ;break status and control register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
BRKE:       equ   7           ;break enable
BRKA:       equ   6           ;break active
; bit position masks
mBRKE:      equ   %10000000   ;break enable
mBRKA:      equ   %01000000   ;break active

BRKH:       equ   $FE0C       ;break address register high
BRKL:       equ   $FE0D       ;break address register low

;****  Low-Voltage Inhibit (LVI)  *******************************************
LVISR:      equ   $FE0F       ;LVI status register
; bit numbers for use in BCLR, BSET, BRCLR, and BRSET
LVIOUT:     equ   7           ;LVI output
LVIIE:      equ   6           ;LVI interrupt enable bit
LVIIF:      equ   5           ;LVI interrupt flag
LVIIAK:     equ   4           ;LVI interrupt request acknowledge bit
; bit position masks
mLVIOUT:    equ   %10000000   ;LVI output
mLVIIE:     equ   %01000000   ;LVI interrupt enable bit
mLVIIF:     equ   %00100000   ;LVI interrupt flag
mLVIIAK:    equ   %00010000   ;LVI interrupt request acknowledge bit

;****  Computer Operating Properly (COP)  ***********************************
COPCTL:     equ   $FFFF       ;COP control register
```

```
;*******************************************************************************
; Registers definition
;*******************************************************************************
; PART I: MCU Related values and regsisters
; ------------------------------------------------------------------------------ *
;
CONFIG1_Init     equ %00100011     ; Config 1 initial (LVI enable not in stop)
;                    |||||||
;                    |||||||+-- COP=disable(1)
;                    ||||||+--- Stop=enable(1)
;                    |||||+---- Recovery time=long(0)
;                    ||||+----- NIL=(0)
;                    |||+------ LVI power=on(0)
;                    ||+------- LVI reset=disable(1)
;                    |+-------- LVI in stop=disable(0)
;                    +--------- COP=long(0)
;
CONFIG2_Init     equ %01111101     ; Config 2 initial
;                    |||||||
;                    |||||||+-- LVI trip point[0] [1:0]= 1:0 = 5.0V
;                    ||||||+--- LVI trip point[1] [1:0]= 0:1 = 3.0V
;                    |||||+---- PTC0-3 = LCD(1)
;                    ||||+----- PTC4-7 = LCD(1)
;                    |||+------ PTD = LCD(1) Need set PTD=$FF (1st 908LT8)
;                    ||+------- PTE = LCD(1) Need set PTE=$FF (1st 908LT8)
;                    |+-------- XTAL enable in stop(1)
;                    +--------- OSC enable in stop(1)
;
LVISR_Init       equ %01010000     ; LVISR initial
;                    |||||||
;                    ||||++++-- NIL=0 (bit 3-0)
;                    |||+------ LVI interrupt acknowledge (1=clear LVI int flag)
;                    ||+------- LVI interrupt flag (1=pending)
;                    |+-------- LVI interrupt enable (1=enable)
;                    +--------- LVI o/p flag(=1 if Vdd fall below LVI trip pt)
;
INTSCR_Init      equ %00000110     ; INTSCR initial
;                    |||||||
;                    |||||||+-- IRQ Edge/Level (1=falling/low level, 0=falling)
;                    ||||||+--- IRQ interrupt mask (1=IRQ interrupt mask)
;                    |||||+---- IRQ interrupt acknowledge (1=clear IRQ int flag)
;                    ||||+----- IRQ interrupt flag (1=pending)
;                    ++++------ NIL=0 (bit 7-4)
;
HDB_Init         equ %00001110     ; HDB initial
;                    |||||||
;                    |||||||+-- PPI clock[0] [1:0]= 10 = 32.768KHz Xtal
;                    ||||||+--- PPI clock[1] [1:0]= 01 = external PPIECLK pin
;                    |||||+---- HDB2 (1= PTB2 is high current)
;                    ||||+----- HDB3 (1= PTB3 is high current)
;                    ||++------ b4-b5: NIL=0 (bit 4 & 5)
;                    |+-------- PPI1 interrupt flag (1=pending)
;                    +--------- NIL=0 (bit 7)
;                    *Remark: PPIECLK mux with PTB2 & T1CH0 !!!!
```

```
;
KBSCR_Init       equ %00000100     ; KBSCR initial
;                    ||||||||
;                    |||||||+-- KBI Edge/Level (1=falling/low level, 0=falling)
;                    ||||||+--- KBI/PPI interrupt mask (1=KBI/PPI int mask)
;                    |||||+---- KBI/PPI interrupt ack (1=clear KBI/PPI int flag)
;                    ||||+----- KBI interrupt flag (1=pending)
;                    ++++------ NIL=0 (bit 7-4)
; * Remark: For KBI int, needs to check KBI interrupt flag in KBSCR
; * Remark: For PPI int, needs to check PPI1 interrupt flag in HDB
;
KBIER_Init       equ %00000011     ; KBIER initial (KBI Enable)
;                    ||||||||
;                    |||||||+-- KBIE0 (1= PTA0 KBI interrupt enable)
;                    ||||||+--- KBIE1 (1= PTA1 KBI interrupt enable)
;                    |||||+---- KBIE2 (1= PTA2 KBI interrupt enable)
;                    ||||+----- KBIE3 (1= PTA3 KBI interrupt enable)
;                    |||+------ PPI1E[0] (000)=PPI disable, (001)= 512 count
;                    ||+------- PPI1E[1] (010)= 1024, (011)= 2048, (100)= 4096
;                    |+-------- PPI1E[2] (101)= 8192, (110)= 16834, (111)= 32768
;                    +--------- NIL=0 (bit 7)
; * Remark: When KBIEx = 1, internal pullup enable and PTAx force to i/p
;
LCDCR_Init       equ %10010000     ; LCDCR initial (LCD Enable)
;                    ||||||||
;                    |||||||+-- LCCON0 (LCCON3:0 is bias voltage control)
;                    ||||||+--- LCCON1
;                    |||||+---- LCCON2
;                    ||||+----- LCCON3
;                    |||+------ LC (LC =1 means low current, FC =1 fast charge)
;                    ||+------- FC (FC:LC) x:0= 37K, 0:1= 146K, 1:1= Fast Charge
;                    |+-------- NIL=0 (bit 6)
;                    +--------- LCDE (1= LCD enable)
;
LCDCLK_Init      equ %00010001     ; LCDCLK initial
;                    ||||||||
;                    |||||||+-- LCLK0 000=256Hz, 001=128Hz, 010=64Hz, 011=32Hz
;                    ||||||+--- LCLK1 Frame rate = LCDCLK * Duty cycle
;                    |||||+---- LCLK2 with LCDCLK=256Hz => 256*(1/4)=64Hz
;                    ||||+----- DUTY[0] 00=Static, 01=1/3 duty cycle
;                    |||+------ DUTY[1] 10=1/4 duty cycle, 11=Not used
;                    ||+------- FCCTL[0] 00=LCDCLK/32, 01=/64, 10=/128 w/LC=0
;                    |+-------- FCCTL[1] (Fast charge duty cycle)
;                    +--------- NIL=0 (bit 7)
; ------------------------------------------------------------------------------ *
;
```

```
; ------------------------------------------------------------------------ *
; PART II: System Related values and variables
; ------------------------------------------------------------------------ *
Auto_Mode_Init   equ %10100000 ;25oC, Sleep_off, swing_off, auto_wind
Heat_Mode_Init   equ %11010000 ;28oC, Sleep_off, swing_off, auto_wind
Data10_Init      equ %00000000 ;AC_OFF, Auto_mode, shift out b0-3
Data32_Init      equ %10100000 ;25oC, Sleep_off, Swing_off, auto_wind
Data54_Init      equ %00001000 ;Model Set to Model 0 (b0 always equal to 0)
Data76_Init      equ %00000010 ;oC, oF low_range,0,health_off,light_on,strong_off
CtmCode_Init     equ %10101001 ;0.63ms low + 0100101 customer code (Tx LSB first)

; Frame range limit for FSL with +/- 200uS
Head_Time_Max    equ $2FA8     ; Header Time Max. (8000+4000)+200 ($2FA8)
Head_Time_Min    equ $2E18     ; Header Time Min. (8000+4000)-200 ($2E18)
Data0_Time_Max   equ $04B0     ; Data0 Time Max. (500+500)+200 ($4B0)
Data0_Time_Min   equ $0320     ; Data0 Time Min. (500+500)-200 ($320)
Data1_Time_Max   equ $0898     ; Data1 Time Max. (500+1500)+200  ($898)
Data1_Time_Min   equ $0708     ; Data1 Time Min. (500+1500)-200  ($708)

; ------------------------------------------------------------------------ *
; Key_Flag bit definition
KEY_ON           equ 7         ;=1 if KBI occur, =0 if key released
AC_OFF_LED       equ 3         ;=0 if AC OFF
AC_ON_LED        equ 2         ;=0 if AC ON

; ------------------------------------------------------------------------ *

DEFAULT_RAM                    SECTION SHORT

;       org       RamStart  ; $0080 ($80-$8F)
Key_Flag         ds  1     ; KEY_ON flag (1=key interrupt occur)

Auto_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%10010000
Cool_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%10010000
Humd_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%10010000
Wind_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%10010000
Heat_Mode        ds  1     ; Store wind speed + Temperature (Tx_Data32)=%11000000

Tx_Data10        ds  1     ; Nibble 1-0 (first 4-bit will be shift out)=%0000xxxx
Tx_Data32        ds  1     ; Nibble 3-2 =%10010000
Tx_Data54        ds  1     ; Nibble 5-4 =%00000001
Tx_CtmCode       ds  1     ; Nibble 9-8 (last bit will be shift out)=%x0100101

Data_Buffer      ds  5     ; Frame Data buffers for Tx_Data10 -- Tx_CtmCode

Byte_Cnt         ds  1     ; Byte Counter for data buffer
Bit_Cnt          ds  1     ; Bit Counter for data buffer

Model_Tmp        ds  1     ; Model Temp
```

```
;*******************************************************************************


DEFAULT_ROM             SECTION

; ------------------------------------------------------------------------------ *
; Program Area                                                                   *
; ------------------------------------------------------------------------------ *
;       org       RomStart  ; $DE00
; ------------------------------------------------------------------------------ *
; Subroutine Initialization : Configure register, Port, KBI & Timer              *
; In      :  <nil>                                                               *
; Out     :  <nil>                                                               *
; Call    :  <nil>                                                               *
; ------------------------------------------------------------------------------ *
;*******************************************************************************
; Entry Point
;*******************************************************************************
Entry:
main:
       rsp                 ; initialize the stack pointer
       SEI                 ; mask all interrupt
       sta $FFFF           ; Clear COP counter
; ------------------------------------------------------------------------------ *
; Port Initial
; ------------------------------------------------------------------------------ *
       mov #%00000011,PTA  ; Avoid false KBI by write PTA0-1 as o/p high
       mov #%11111111,DDRA ; Set all port to output low except PTA0-1
                           ; Set PTA3-7 o/p low, KBI will occur if any key
                           ; pressed after KBI interrupt enable

       clr PTC             ; Set all port to output low
       clr PTD
       clr PTE
       clr PTB

       lda #$FF            ; Change port data register before change data
       sta DDRB            ; direction register (aviod glitch)
       sta DDRC
       sta DDRD
       sta DDRE
       sta PTD             ; enable LCD output
       sta PTE             ; enable LCD output
;*******************************************************************************
; ------------------------------------------------------------------------------ *
; Configuration Register Initial
; ------------------------------------------------------------------------------ *
       mov #CONFIG1_Init,CONFIG1   ; CONFIG1 Initial
       mov #CONFIG2_Init,CONFIG2   ; CONFIG2 Initial
       lda #LVISR_Init
       sta LVISR                   ; LVI Initial
       mov #INTSCR_Init,INTSCR     ; IRQ Initial
```

```
; ------------------------------------------------------------------------ *
; Keyboard Interrupt & PPI Initial
; ------------------------------------------------------------------------ *
       mov #HDB_Init,HDB           ; PTB & PPI Clock Initial
       mov #KBSCR_Init,KBSCR       ; KBI & PPI Initial
       mov #KBIER_Init,KBIER       ; KBI & PPI Interrupt Initial
; ------------------------------------------------------------------------ *
; Keyboard status and Mode & Tx_Data Initial
; ------------------------------------------------------------------------ *
       clr Key_Flag            ; Initial Keyboard flag & value and counter

       clrh
       clrx
Clr_Buffer:
       clr Data_Buffer,x       ; Initial Frame data buffer register
       incx
       cpx  #$04
       blo  Clr_Buffer

       lda #Auto_Mode_Init     ; Initial difference mode value
                               ; wind speed + Temperature (Tx_Data32)
       sta Auto_Mode
       sta Cool_Mode
       sta Humd_Mode
       sta Wind_Mode
       mov #Heat_Mode_Init,Heat_Mode

       mov #Data10_Init,Tx_Data10   ; Initial Tx Data + Customer code
       mov #Data32_Init,Tx_Data32
       mov #Data54_Init,Tx_Data54
       mov #CtmCode_Init,Tx_CtmCode

; ------------------------------------------------------------------------ *
; LCD Data & Configuration Register Initial
; ------------------------------------------------------------------------ *
       clrh
       clrx
       lda #$FF                ; default LCD pattern (All ON)
Ld_LCD_FF:
       sta LDAT1,x
       incx
       cpx #$0D
       blo Ld_LCD_FF

       mov #%10101111,LDAT9    ; OFF unused LCD segments
       mov #%10000000,LDAT5
       bclr 4,LDAT4

       mov #LCDCR_Init,LCDCR   ; Initial LCD
       mov #LCDCLK_Init,LCDCLK

       jsr Delay_500ms         ; All segment in LCD panel ON (1 sec)
       jsr Delay_500ms
```

```
; ------------------------------------------------------------- *
       clrh
       clrx
       lda #$00                ; default LCD pattern(All OFF except 25oC + Light)
Ld_LCD_00:
       sta LDAT1,x
       incx
       cpx #$0A                ; Reach to LDAT11?
       bls Ld_LCD_00

       bclr AC_OFF_LED,PTB     ; Turn ON "AC_OFF" LED
       bset AC_ON_LED,PTB      ; Turn OFF "AC_ON" LED

; ------------------------------------------------------------------------- *
; Remote Run after power on reset (POR) --- Waiting for Keyboard Interrupt
; ------------------------------------------------------------------------- *
Main_Loop:

; Set T2CH0 as input capture to decode the IR receiving signal

      mov #%00110000,T2SC     ; TOF int disable, Timer stop & reset, clock=bus/1
      mov #%01001000,T2SC0    ; Int enable, input capture, falling edges


      CLI                     ; Clear interrupt mask bit

                              ; Enable TIM2 Now
      bclr TSTOP,T2SC         ; clear TSTOP, enable timer counter
      bclr CH0F,T2SC0         ; clear CH0F flag
      bclr TOF,T2SC           ; clear TOF flag

Frame_Restart:

      clr Bit_Cnt
      clr Byte_Cnt

      wait                    ; waiting for IR signal OR Key pressed

      brclr KEY_ON,Key_Flag,No_K_Scan   ; Receive IR frame signal if No key

      jmp  K_Scan             ; Perfrom key scan if KBI occurred

No_K_Scan:

      wait                    ; wait for second falling edge of header

      ldhx  T2CH0H            ; load T2CH0H to H and T2CH0L to X
      cphx  #Head_Time_Max
      bhi   Frame_Restart
      cphx  #Head_Time_Min
      blo   Frame_Restart
; 8ms + 4ms (+/- 200uS) "Header" is O.K.

Rep_Data:                     ; Here Header is passed
```

```
      wait                    ; Waiting for 1st data bit

Tst_Data0:
      ldhx  T2CH0H            ; Store Frame counter to H:X
      cphx  #Data0_Time_Max
      bhi   Tst_Data1
      cphx  #Data0_Time_Min
      blo   Frame_Error
      bsr   St_Data0
; 500uS + 500us (+/- 200uS) "Data0" is O.K.

      bra   Tst_Next_Data

Tst_Data1:
      ldhx  T2CH0H            ; Store Frame counter to H:X
      cphx  #Data1_Time_Max
      bhi   Frame_Error
      cphx  #Data1_Time_Min
      blo   Frame_Error
      bsr   St_Data1
; 500uS + 1500us (+/- 200uS) "Data1" is O.K.

Tst_Next_Data:

      lda Byte_Cnt
      cmp #$04                ; Is all Data received ?
      blo Rep_Data

      bra St_Data_End

Frame_Error:
                              ; Insert any action in here if needed

      bra  Frame_Restart      ; Restart Frame if error occurred
;---------------------------------------------------------------
St_Data0:                     ; Store Data0 to buffer

      clrh
      ldx   Byte_Cnt
      clc
      ror   Data_Buffer,x     ; X = Byte_Cnt
      inc   Bit_Cnt
      lda   Bit_Cnt
      cmp   #$08
      blo   Bit_Cnt_No_Inc0
      clr   Bit_Cnt
      inc   Byte_Cnt
Bit_Cnt_No_Inc0:
      rts
;---------------------------------------------------------------
St_Data1:                     ; Store Data1 to buffer

      clrh
```

```
      ldx   Byte_Cnt
      sec
      ror   Data_Buffer,x     ; X = Byte_Cnt
      inc   Bit_Cnt
      lda   Bit_Cnt
      cmp   #$08
      blo   Bit_Cnt_No_Inc1
      clr   Bit_Cnt
      inc   Byte_Cnt
Bit_Cnt_No_Inc1:
      rts
;--------------------------------------------------------------

St_Data_End:

      clrh
      ldx   #$02
      lda   Data_Buffer,x     ; Load Data_Buffer(Tx_Data54)
      and   #%11110000        ; Mask all bit except Model #
      sta   Model_Tmp

      lda   Tx_Data54
      and   #%11110000        ; Mask all bit except Model #
      cmp   Model_Tmp

      bne   Frame_Error       ; Restart Rx frame if model wrong


            ldx   #$03
      lda   Data_Buffer,x     ; Load Data_Buffer(Tx_CtmCode)
      cmp   #CtmCode_Init
      bne   Frame_Error       ; Restart Rx frame if cmt code wrong


      clrh                    ; Here Model & customer code are O.K.
      clrx
Update_Buffer:
      lda   Data_Buffer,x     ; Update Frame from Data buffer
      sta   Tx_Data10,x
      incx
      cpx   #$04
      blo   Update_Buffer

Key_Action:

      bclr  KEY_ON,Key_Flag   ; Clear KEY_ON flag

      jsr   LCD_Update        ; Update LCD display

      jmp   Frame_Restart
```

```
; ------------------------------------------------------------------------- *
; Keyboard Scan Routine                                                     *
; In      :  <Key_Flag>                                                     *
; Out     :  <Key_Value>                                                    *
; Call    :  <nil>                                                          *
; ------------------------------------------------------------------------- *
K_Scan:                         ; Key location and expect KEY_ON=1

       SEI                      ; Interrupt mask enable

       lda #$0A                 ; 10*0.5ms
       jsr Delay_Hms            ; Delay 5ms for Key Debounce

       mov #%11111100,DDRA      ; Set PTA0 & PTA1 as i/p

       brset 0,PTA,Chk_PTA1

Chk_PTA0:
       lda #$0A                 ; 10*0.5ms
       jsr Delay_Hms            ; Delay 5ms for Key Debounce

       brset 0,PTA,Bad_Key

       brclr 7,Tx_Data10,SW_ON  ; A/C OFF

       bclr  7,Tx_Data10        ; A/C ON => A/C OFF

       bra  Key_Action

SW_ON:
       bset 7,Tx_Data10         ; A/C OFF => A/C ON

       bra  Key_Action

Chk_PTA1:

       brset 1,PTA,Bad_Key

       lda #$0A                 ; 10*0.5ms
       jsr Delay_Hms            ; Delay 5ms for Key Debounce

       brset 1,PTA,Bad_Key

       clrh
       lda Tx_Data54
       nsa
       and #%00001111
       tax
       cpx #$09                 ; Reach Max Model 9?
       blo Inc_Model            ; Lower than Model 9
       clrx                     ; Reset Timer Pointer to Min(Model 0)
       bra Dec_Model
Inc_Model:
       incx                     ; Increase Timer setting
```

```
Dec_Model:
       lda $F030,x             ; Load Data54 Table
       sta Tx_Data54           ; Update Data54 (4.3=1)

Bad_Key:

       CLI                     ; Interrupt mask disable

       bra  Key_Action

; ------------------------------------------------------------------------- *
; Subroutine <500ms Delay>                                                  *
; Bus Clock = 1MHz, 1 Cycle=1uS                                             *
; In      :  <nil>                                                          *
; Out     :  <nil>                                                          *
; Call    :  <nil>                                                          *
; ------------------------------------------------------------------------- *
Delay_500ms:
       LDX #$C8                ; [2]200
Delay_Yms_X
       LDA #$FA                ; [2]250
Delay_Yms_A
       sta $FFFF               ; [4] clear COP
       nop                     ; [1]
       nop                     ; [1]
       nop                     ; [1]
       dbnza Delay_Yms_A       ; [3] 10*{A}=10A
       dbnzx Delay_Yms_X       ; [3] 10A*{X}+2+3=10AX+2+3
       rts                     ; [3]

;Total= {[2]+[2]+10AX+[3]}*Bus Cycle
;Total= [7+(10*200*250)]*1uS = 500mS

; ------------------------------------------------------------------------- *
; Subroutine <Hms Delay>                                                    *
; Bus Clock = 1MHz, 1 Cycle=1uS                                             *
; In      :  Acc                                                            *
; Out     :  <nil>                                                          *
; Call    :  <nil>                                                          *
; Remark  : (0.5mS * A) Delay                                               *
; ------------------------------------------------------------------------- *
Delay_Hms:

Delay_Hms_A:
       LDX #$31                ; [2]49
Delay_Hms_X:
       sta $FFFF               ; [4] clear COP
       nop                     ; [1]
       nop                     ; [1]
       nop                     ; [1]
       dbnzx Delay_Hms_X       ; [3] 10*{X}=10X
       dbnza Delay_Hms_A       ; [3] 10X*{A}+2+3=10AX+5
       rts                     ; [3] 2+10XA+5+3=10XA+10
```

```
;Total= {10AX+[10]}*Bus Cycle
;Total= [10+(10*49*A)]*1uS = 500uS*A =A*0.5mS
; ---------------------------------------------------------------------------

; ------------------------------------------------------------------------- *
; LCD Data Update Subroutine (Depend on key value and then Tx Data)
; ------------------------------------------------------------------------- *
LCD_Update:

       jsr LCD_N2_Chk          ; LCD based on N2
       jsr LCD_N3_Chk          ; LCD based on N3
       jsr LCD_N45_Chk         ; LCD based on N4 & N5
       jsr LCD_N67_Chk         ; LCD based on N6 & N7
       jsr LCD_N1_Chk          ; LCD based on N1
       jsr LCD_N2_Sleep        ; ALL LCD OFF if sleep mode enable

       rts

; ------------------------------------------------------------------------- *
LCD_N1_Chk:
       brclr 7,Tx_Data10,L_AC_OFF
                               ; Here AC ON

       bclr AC_ON_LED,PTB      ; Turn on "AC_ON" LED
       bset AC_OFF_LED,PTB     ; Turn off "AC_OFF" LED

       lda Tx_Data10
       and #%01110000          ; Mask all bit except bit4-6
       cmp #%00000000          ; Is bit4-6 = %000
       beq LCD_Auto_Mode       ; Go to Auto mode if true
       cmp #%00010000          ; Is bit4-6 = %001
       beq LCD_Cool_Mode       ; Go to Cool mode if true
       cmp #%00100000          ; Is bit4-6 = %010
       beq LCD_Humd_Mode       ; Go to Humd mode if true
       cmp #%00110000          ; Is bit4-6 = %011
       beq LCD_Wind_Mode       ; Go to Wind mode if true
LCD_Heat_Mode:                 ; Otherwise go to Heat mode (%100 - %111)
       bclr 3,LDAT10           ; AUTO OFF
       bclr 1,LDAT10           ; COOL OFF
       bclr 0,LDAT10           ; DRY OFF
       bclr 7,LDAT9            ; WIND OFF
       bset 2,LDAT10           ; HEAT ON
       rts
LCD_Auto_Mode:
       bset 3,LDAT10           ; AUTO ON
       bclr 1,LDAT10           ; COOL OFF
       bclr 0,LDAT10           ; DRY OFF
       bclr 7,LDAT9            ; WIND OFF
       bclr 2,LDAT10           ; HEAT OFF
       rts
LCD_Cool_Mode:
       bclr 3,LDAT10           ; AUTO OFF
       bset 1,LDAT10           ; COOL OFF
       bclr 0,LDAT10           ; DRY OFF
```

```
        bclr 7,LDAT9            ; WIND OFF
        bclr 2,LDAT10           ; HEAT ON
        rts
LCD_Humd_Mode:
        bclr 3,LDAT10           ; AUTO OFF
        bclr 1,LDAT10           ; COOL OFF
        bset 0,LDAT10           ; DRY OFF
        bclr 7,LDAT9            ; WIND OFF
        bclr 2,LDAT10           ; HEAT ON
        rts
LCD_Wind_Mode:
        bclr 3,LDAT10           ; AUTO OFF
        bclr 1,LDAT10           ; COOL OFF
        bclr 0,LDAT10           ; DRY OFF
        bset 7,LDAT9            ; WIND OFF
        bclr 2,LDAT10           ; HEAT ON
        rts

L_AC_OFF:

        bsr  LCD_All_OFF

        bset AC_ON_LED,PTB       ; Turn off "AC_ON" LED
        bclr AC_OFF_LED,PTB      ; Turn on "AC_OFF" LED

        rts
; -------------------------- *
LCD_N2_Sleep:

        brclr 3,Tx_Data32,L_Sleep_OFF
LCD_All_OFF:
        clrh
        clrx
        clra                        ; ALL LCD OFF if sleep mode
Ld_LCD_OFF:
        sta LDAT1,x
        incx
        cpx #$0D
        blo Ld_LCD_OFF

L_Sleep_OFF:

         rts
; ------------------------------------------------------------------------- *
LCD_N2_Chk:
; -------------------------- *
LCD_N2_Swing:
        brclr 2,Tx_Data32,L_Swing_OFF
        brclr 7,Tx_Data10,L_Swing_OFF ; Swing OFF if AC OFF

                                      ; Here AC ON

        bset 3,LDAT3                  ; A.M.WIND ON
        bra LCD_N2_Wind_Speed
```

```
L_Swing_OFF:
       bclr 3,LDAT3                    ; A.M.WIND OFF
; -------------------------- *
LCD_N2_Wind_Speed:

       brclr 7,Tx_Data10,L_Wind_OFF ; Wind Speed OFF if AC OFF
                                    ; Here AC ON
       lda Tx_Data32
       and #%00000011          ; Mask all except bit0-1
       cmp #%00000000          ; Is Bit0-1 = %00
       beq L_Wind_Auto         ; Go to Wind Auto if true
       cmp #%00000001          ; Is Bit0-1 = %01
       beq L_Wind_Low          ; Go to Wind Low if true
       cmp #%00000010          ; Is Bit0-1 = %10
       beq L_Wind_Mid          ; Go to Wind Mid if true
L_Wind_High:                   ; Otherwise go to Wind High (%11)
       bset 4,LDAT3            ; Max Wind ON
       mov #%00000111,LDAT6    ; AUTO OFF, FAN, Min Wind & Mid Wind ON
       rts
L_Wind_Low:
       bclr 4,LDAT3            ; Max Wind OFF
       mov #%00000110,LDAT6    ; FAN & Min Wind ON, AUTO & Mid Wind OFF
       rts
L_Wind_Mid:
       bclr 4,LDAT3            ; Max Wind OFF
       mov #%00000111,LDAT6    ; FAN & Min Wind & Mid Wind ON, AUTO OFF
       rts
L_Wind_Auto:
       bclr 4,LDAT3            ; Max Wind OFF
       mov #%00001100,LDAT6    ; FAN & AUTO ON, Min Wind & Mid Wind OFF
       rts

L_Wind_OFF:
       bclr 4,LDAT3
       clr LDAT6               ; All OFF if AC OFF
       rts
; ------------------------------------------------------------------------------ *
LCD_N3_Chk:

       brclr 7,Tx_Data10,L_N3_AC_OFF

L_N3_AC_ON:                    ; Here AC ON
       lda Tx_Data10
       and #%01110000          ; Mask all bit except bit4-6
       cmp #%00000000          ; Is bit4-6 = %000
       beq L_N3_Auto_oC        ; Go to Auto mode if true
       bra L_N3_oC_ON

L_N3_Auto_oC:
       clr LDAT7               ; XXoC OFF in AUTO mode
       clr LDAT8
       rts
```

```
L_N3_AC_OFF:                    ; Here AC OFF
L_N3_oC_ON:

       clrh
       lda Tx_Data32
       and #%11110000
       nsa
       tax                      ; Store N3 as Index
       lda $F000,x
       sta LDAT8
       aix #$10
       lda $F000,x
       sta LDAT7
       rts
; ---------------------------------------------------------------------------- *
LCD_N45_Chk:

       brset 3,Tx_Data54,L_MODEL_ON  ; Flash if Model Set

L_MODEL_OFF:
       bclr 7,LDAT5                  ; MODEL ON to OFF
       bra L_7SEG_CHK

L_MODEL_ON:
       bset 7,LDAT5                  ; MODEL ON to OFF
       bra L_7SEG_CHK
; -------------------------- *
L_7SEG_CHK:

       clrh                     ; Update 5.3-5.0 7-Segment
       lda Tx_Data54
       nsa                      ; swrap N5 to lower nibble
       and #%00001111           ; mask N4 (Force bit 4-7 =0)
       tax
       lda $F020,x              ; Load 7-segment data
       sta LDAT4                ; Display x(0-9)
; -------------------------- *

       bset 7,LDAT5             ; MODEL ON

       rts
; ---------------------------------------------------------------------------- *
LCD_N67_Chk:                          ; Only 6.1 need to check
       brclr 1,Tx_Data10,L_Light_OFF
       bset 0,LDAT3                   ; "light" ON (M.WIND)
       bra L_Light_END
L_Light_OFF:
       bclr 0,LDAT3                   ; "light" OFF (M.WIND)
L_Light_END:

       brclr 7,Tx_Data10,L_N7_AC_OFF  ; Check AC ON/OFF?
```

```
L_N7_AC_ON:
       lda Tx_Data10
       and #%01110000                  ; mask off except 1.2, 1.1, 1.0
       beq L_C_OFF

       bra L_N7_AC_OFF

L_C_OFF:
       bclr 4,LDAT8                    ; oC OFF if in AC ON & Auto mode
       bra L_N67_END

L_N7_AC_OFF:
       bset 4,LDAT8                    ; oC ON if AC OFF
L_N67_END:
       rts

; ------------------------------------------------------------------------------ *
; Program Interrupt Service Routine Area                                         *
; ------------------------------------------------------------------------------ *
;      org       $EE00
; ------------------------------------------------------------------------------ *
; DMY_ISR - dummy Interrupt Service Routine (with no operation)                  *
; In      :  <nil>                                                               *
; Out     :  <nil>                                                               *
; Call    :  <nil>                                                               *
; ------------------------------------------------------------------------------ *
DMY_ISR:
       nop
       rti
; ------------------------------------------------------------------------------ *
; Timer Interrupt Service Routine                                                *
; In      :  <nil>                                                               *
; Out     :  <nil>                                                               *
; Call    :  <nil>                                                               *
; ------------------------------------------------------------------------------ *
T1M0_ISR:                       ; Unused
       bclr CH0F,T1SC0          ; clear CH0F flag in TIM1
       rti
; ------------------------------------------------------------------------------ *
T2M0_ISR:                       ; As TIM2 Input capture

       mov  #%00110000,T2SC     ; TOF int disable, Timer stop & reset, clock=bus/1
       bclr TSTOP,T2SC          ; clear TSTOP, enable timer counter
       bclr CH0F,T2SC0          ; clear CH0F flag
       bclr TOF,T2SC

       rti
; ------------------------------------------------------------------------------ *
T1M1_ISR:                       ; Unused
       bclr CH1F,T1SC1          ; clear CH1F flag in TIM1
       rti
; ------------------------------------------------------------------------------ *
```

```
T2M1_ISR:                       ; Unused
      bclr CH1F,T2SC1           ; clear CH1F flag in TIM2
      rti
; ------------------------------------------------------------------------- *
T1OF_ISR:                       ; Unused
       bclr TOF,T1SC            ; clear TIM1 TOF flag in TIM1

       rti
; ------------------------------------------------------------------------- *
T2OF_ISR:                       ; Unused
      bclr TOF,T2SC             ; clear TIM2 TOF flag in TIM2

      rti
; ------------------------------------------------------------------------- *
; KBI_ISR - Keyboard & PPI Interrupt Service Routine                        *
; In      :  <nil>                                                          *
; Out     :  <nil>                                                          *
; Call    :  <nil>                                                          *
; Remark  : Either PPI or KBI enable at the same time                       *
; ------------------------------------------------------------------------- *
KBI_ISR:

     lda PTA                    ; Toggles PTA2 when KBI interrupt present
     eor #%00000100             ; For Debug
     sta PTA

PPI_CHK:
     brset PPI1L,HDB,PPI_ACK    ; check PPI interrupt flag
     bra KBI_ACK                ; If not a PPI interrupt, jump to KBI_ACK

PPI_ACK:
     bset ACKK,KBSCR            ; clear PPI1L flag by set ACKK bit
     brset PPI1L,HDB,PPI_ACK    ; check PPI int flag clear

     rti

KBI_ACK:
     bset KEY_ON,Key_Flag       ; Set KEY_ON flag
     bset ACKK,KBSCR            ; clear PPI1L flag by set ACKK bit

     rti
; ------------------------------------------------------------------------- *
; LVI_ISR - LVI Interrupt Service Routine                                   *
; In      :  <nil>                                                          *
; Out     :  <nil>                                                          *
; Call    :  <nil>                                                          *
; ------------------------------------------------------------------------- *
LVI_ISR:
       lda #%01010000
       sta LVISR                ; clear LVI int. flag (Cannot use bset 4,LVISR)
       rti                      ; Cannot use "bset 4,LVISR" due to LVISR = $FE0F)
; ------------------------------------------------------------------------- *
; IRQ_ISR - IRQ Interrupt Service Routine                                   *
; In      :  <nil>                                                          *
```

```
; Out     :  <nil>                                                          *
; Call    :  <nil>                                                          *
; ------------------------------------------------------------------------ *
IRQ_ISR:
       bset 2,INTSCR           ; CLEAR IRQF
       rti

; ------------------------------------------------------------------------ *
; Program Look Up Table Area (For LCD #1)                                   *
; ------------------------------------------------------------------------ *
       org       $F000         ;$F000-$F01F
; ------------------------------------------------------------------------ *
; LCD oC Display Lookup Table

;           ABC%XGED   ; 10th digit of oC (%=oC & TEMP always ON, X always OFF)
       FCB %01110000   ; Digit 1 (B,C)          [X=$00]
       FCB %01110000   ; Digit 1 (B,C)          [X=$01]
       FCB %01110000   ; Digit 1 (B,C)          [X=$02]
       FCB %01110000   ; Digit 1 (B,C)          [X=$03]
       FCB %01110000   ; Digit 1 (B,C)          [X=$04]

       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$05]
       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$06]
       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$07]
       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$08]
       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$09]
       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$0A]
       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$0B]
       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$0C]
       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$0D]
       FCB %11010111   ; Digit 2 (A,B,D,E,G)    [X=$0E]

       FCB %11110101   ; Digit 3 (A,B,C,D,G)    [X=$0F]
; ------------------------------------------------------------------ *
;           ABCXFGED   ; 1st digit of oC (X always OFF)
       FCB %10101101   ; Digit 5 (A,C,D,F,G)       [X=$00+$10]
       FCB %10101111   ; Digit 6 (A,C,D,E,F,G)     [X=$01+$10]
       FCB %11100000   ; Digit 7 (A,B,C)           [X=$02+$10]
       FCB %11101111   ; Digit 8 (A,B,C,D,E,F,G)   [X=$03+$10]
       FCB %11101101   ; Digit 9 (A,B,C,D,F,G)     [X=$04+$10]

       FCB %11101011   ; Digit 0 (A,B,C,D,E,F)     [X=$05+$10]
       FCB %01100000   ; Digit 1 (B,C)             [X=$06+$10]
       FCB %11000111   ; Digit 2 (A,B,D,E,G)       [X=$07+$10]
       FCB %11100101   ; Digit 3 (A,B,C,D,G)       [X=$08+$10]
       FCB %01101100   ; Digit 4 (B,C,F,G)         [X=$09+$10]
       FCB %10101101   ; Digit 5 (A,C,D,F,G)       [X=$0A+$10]
       FCB %10101111   ; Digit 6 (A,C,D,E,F,G)     [X=$0B+$10]
       FCB %11100000   ; Digit 7 (A,B,C)           [X=$0C+$10]
       FCB %11101111   ; Digit 8 (A,B,C,D,E,F,G)   [X=$0D+$10]
       FCB %11101101   ; Digit 9 (A,B,C,D,F,G)     [X=$0E+$10]

       FCB %11101011   ; Digit 0 (A,B,C,D,E,F)     [X=$0F+$10]
```

```
; ------------------------------------------------------------------------- *
      org       $F020         ;$F020-$F029
; ------------------------------------------------------------------------- *
; Model Number Setting Display Lookup Table

;           ABCXFGED   ;
      FCB %11101011   ; Digit 0 (A,B,C,D,E,F)    [X=$00]
      FCB %01100000   ; Digit 1 (B,C)            [X=$01]
      FCB %11000111   ; Digit 2 (A,B,D,E,G)      [X=$02]
      FCB %11100101   ; Digit 3 (A,B,C,D,G)      [X=$03]
      FCB %01101100   ; Digit 4 (B,C,F,G)        [X=$04]
      FCB %10101101   ; Digit 5 (A,C,D,F,G)      [X=$05]
      FCB %10101111   ; Digit 6 (A,C,D,E,F,G)    [X=$06]
      FCB %11100000   ; Digit 7 (A,B,C)          [X=$07]
      FCB %11101111   ; Digit 8 (A,B,C,D,E,F,G)  [X=$08]
      FCB %11101101   ; Digit 9 (A,B,C,D,F,G)    [X=$09]

; ------------------------------------------------------------------------- *
      org       $F030         ;$F030-$F039
; ------------------------------------------------------------------------- *
; Key Press Model Setting Lookup Table

; from Model 0(x=00) to Model 9 (X=09);
; ------------------------------------------- *

      FCB %00001000   ; 0 Model     [X=$00]
      FCB %00011000   ; 1 Model     [X=$01]
      FCB %00101000   ; 2 Model     [X=$02]
      FCB %00111000   ; 3 Model     [X=$03]
      FCB %01001000   ; 4 Model     [X=$04]
      FCB %01011000   ; 5 Model     [X=$05]
      FCB %01101000   ; 6 Model     [X=$06]
      FCB %01111000   ; 7 Model     [X=$07]
      FCB %10001000   ; 8 Model     [X=$08]
      FCB %10011000   ; 9 Model     [X=$09]

; ------------------------------------------------------------------------- *
```

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics of their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to http://www.freescale.com/epp.

DRM082
Rev. 0, 09/2006

freescale™
semiconductor